PROTOTIPO PARA EL MONITOREO DE ESTRUCTURAS EN SISTEMA DE PÓRTICOS.

CÉSAR ARMANDO HERNÁNDEZ PUENTES.

UNIVERSIDAD PILOTO DE COLOMBIA
FACULTAD DE INGENIERÍA DE TELECOMUNICACIONES
PROGRAMA DE INGENIERÍA DE TELECOMUNICACIONES
BOGOTÁ D.C
2016

PROTOTIPO PARA EL MONITOREO DE ESTRUCTURAS EN SISTEMA DE PÓRTICOS.

CÉSAR ARMANDO HERNÁNDEZ PUENTES.

Proyecto de grado para optar al título de Ingeniería de Telecomunicaciones

Directora:
Ing. Ana María Cagua.

UNIVERSIDAD PILOTO DE COLOMBIA
FACULTAD DE INGENIERÍA DE TELECOMUNICACIONES
PROGRAMA DE INGENIERÍA DE TELECOMUNICACIONES
BOGOTÁ D.C
2016

Nota de aceptación:

_____

_____

_____

_____

_____

_____

_____

Firma del presidente del jurado

_____

Firma del jurado

_____

Firma del jurado

Bogotá D.C, 12 de Diciembre de 2016

DEDICATORIA

A mi madre por su infinita tenacidad, a mi hermano por su apoyo y gran conocimiento. A mi mascota que con su naturalidad demostró la fuerza que necesitamos para continuar a pesar de las circunstancias.

AGRADECIMIENTOS

A mi madre por darme ánimo, amor y apoyo para llevar a cabo mis estudios, por mostrarme los valores éticos y morales que me han guiado durante mi vida.

A mi hermano quien participo en la concepción de la idea original y comprobación del proyecto en sí.

Al comité de investigación y a la Ingeniera Ana María Cagua por apoyarme en el desarrollo de este proyecto.

CONTENIDO.

LISTA DE TABLAS

LISTA DE FIGURAS

LISTA DE ANEXOS

GLOSARIO

DOMÓTICA: el término domótica, para algunos, nacido de la combinación de las palabras electrónica y doméstico, y para otros, procedente del latín domus, que significa casa, es en realidad la denominación de una nueva disciplina que no busca más que utilizar la electrónica, la informática y las telecomunicaciones para aumentar la comodidad y sobre todo la seguridad.

TELEMETRÍA: Conjunto de técnicas para la medida a distancia de magnitudes físicas.

INTERNET DE LAS COSAS: en inglés, *Internet of things*, (abreviado *IoT*), es un concepto que se refiere a la interconexión digital de objetos cotidianos con Internet.

SENSOR: dispositivos electrónicos que permiten interactuar con el entorno, de forma que proporcionan información de ciertas variables para procesarlas y así generar o activar procesos.

FIRMWARE: el firmware es un programa informático que establece la lógica de más bajo nivel que controla los circuitos electrónicos de un dispositivo de cualquier tipo.

MICROCONTROLADOR ARDUINO: Arduino es en realidad tres cosas: Una placa de hardware libre que incorpora un microcontrolador programable y una serie de pines hembra (los cuales están unidos internamente a las pastillas de E/S del microcontrolador) que permiten conectar allí de forma muy sencilla y cómoda diferentes sensores y actuadores.

NODEMCU: es una plataforma IoT de código abierto. Incluye el firmware que se ejecuta en el ESP8266.

I2C: El bus I2C fue diseñado por Philips a principios de los años 80 para permitir una fácil comunicación entre los componentes que se encuentran en la misma placa de circuito

ESP8266: es un chip Wi-Fi de bajo costo con la pila TCP/IP completa y la capacidad de microcontrolador

MATLAB: es un lenguaje de cálculo técnico de alto nivel y un entorno interactivo para el desarrollo de algoritmos, la visualización de datos, el análisis de información.

DESPLAZAMIENTOS HORIZONTALES DE LA ESTRUCTURA: Son los desplazamientos horizontales de los entrepisos en la dirección bajo estudio provocados por un sismo.

ASENTAMIENTOS: En general, todos los suelos se compactan o endurecen, es decir se produce un movimiento hacia abajo denominado asentamiento.

SISTEMA APORTICADO: es aquel cuyos elementos estructurales principales consisten en vigas y columnas conectados a través de nudos formando pórticos resistentes en las dos direcciones principales de análisis (x e y).

RESUMEN

El presente proyecto comprende el “Diseño de un dispositivo para medir los desplazamientos horizontales y verticales de una estructura en sistema de pórticos”.

El prototipo se diseñó con la idea de realizar un trabajo interdisciplinar entre la Ingeniería de Telecomunicaciones y la Civil para apoyar a ésta última en la interventoría constante y automatizada a través de las telecomunicaciones mediante la medida de las variables de desplazamiento referidas previamente, y que afectan la resistencia al sistema estructural en los edificios.

El prototipo adquiere a través de un acelerómetro de tres ejes y ultrasonidos magnitudes físicas tales cómo, las aceleraciones en X, Y y Z, la inclinación en X y Y y la distancia en Z a través de un microcontrolador Arduino que transmite los datos inalámbricamente por medio de una conexión Wi-Fi hacia *Thingspeak* en internet, la cual es una plataforma de datos abierta que permite recopilar información de los sensores en la nube.

La información de las mediciones adquiridas es visualizada en dicha plataforma a través de su interfaz gráfica y desde cualquier navegador web, de una forma simple y efectiva. De igual forma, permite el tratamiento de la información mediante MATLAB para obtener cálculos a partir de ella.

Adicionalmente, a través de *Thingspeak* también se produce un análisis trivial de la información para generar alertas que permitan realizar un seguimiento por parte de los expertos de Ingeniera Civil en construcciones basadas en sistemas de pórticos.

INTRODUCCIÓN.

En la actualidad, la construcción de edificaciones es uno de los campos de mayor crecimiento del país. Según el DANE[1], los Indicadores Económicos Alrededor de la Construcción- IEAC en el primer trimestre de 2016 mostraron que el valor agregado del sector de la construcción aumentó 5,2 %, este resultado se explica por el aumento de 10,9 % en el subsector de edificaciones y el incremento de 0,4 % en el subsector de obras civiles.

Colombia hace parte de los países donde el riesgo sísmico es alto dado que se encuentra en el cinturón de fuego del Pacífico donde convergen las placas Nazca, Suramericana, Caribe y la de Cocos. También el suelo posee diferentes características que pueden afectar las edificaciones.

La edificación está regida por la Norma Colombiana para Construcción Sismo Resistente NSR-10 cuyo primer objeto es "*Reducir a un mínimo el riesgo de la pérdida de vidas humanas, y defender en lo posible el patrimonio del Estado y de los ciudadanos.*"[2]

El no seguir los lineamientos descritos en dicha norma implica casos como el presentado en la ciudad de Medellín, donde el colapso del edificio Space dejó 12 personas muertas.

Es allí cuando el trabajo interdisciplinar de las ciencias de la ingeniería toma mayor relevancia, entregando técnicas de medida y trazabilidad de las variables que pueden llegar a afectar dichas estructuras.

La ingeniería de telecomunicaciones interviene allí convirtiéndose en núcleo de la oportunidad y eficiencia a la hora de la entrega de resultados a grandes velocidades, con sistemas de apoyo y medición en tiempo real.

[1] Comunicado de prensa, DANE, Bogotá, 30 de junio de 2016.

[2] NSR-10 objeto A.1.2.2.1, Título A - Requisitos generales de diseño y construcción sismo resistente, p. 1.

1.PLANTEAMIENTO DEL PROYECTO.

1.1 PLANTEAMIENTO DEL PROBLEMA

Las estructuras en sistema de pórticos (

Figura 1) “es un sistema estructural compuesto por un pórtico espacial, resistente a momentos, esencialmente completo, sin diagonales, que resiste todas las cargas verticales y fuerzas horizontales[3]”.

Están sujetos a fenómenos de la naturaleza tales como los sismos y los asentamientos del terreno en donde se encuentran construidas, y adicional a esto se encuentran las fuerzas ejercidas por el uso normal de las mismas.

El soporte de dicho sistema son las columnas, las cuales generan desplazamientos horizontales y verticales que determinarán si la estructura es sismo resistente, si el terreno en donde está construida es estable y si el propósito de la edificación fue contemplado correctamente.

Para la carga horizontal existe un concepto llamado deriva, las derivas son los desplazamientos relativos entre dos pisos consecutivos de una estructura, al someter una estructura a la acción de las cargas sísmicas, los entrepisos (pisos de la estructura) sufren desplazamientos horizontales, cuya magnitud debe limitarse, entre mayor sea su valor mayores serán los daños[4].

De igual forma, durante y posterior a la construcción de las edificaciones, puede llegar a existir negligencia humana en el cumplimiento de las recomendaciones establecidas en la NSR-10, aumentando la vulnerabilidad estructural del edificio por las fuerzas antes mencionadas y así el riesgo de presentar colapso o degradaciones graves que deben ser tomadas a consideración oportunamente.

[3] AWAD, Roberto Rochel. Análisis y diseño sísmico de edificios. Medellín: Fondo editorial Universidad EAFIT, 2012, p. 63.

[4] AWAD, Roberto Rochel. Análisis y diseño sísmico de edificios. Medellín: Fondo editorial Universidad EAFIT, 2012, p. 96.

¿Será posible a través de las telecomunicaciones realizar el monitoreo preventivo a las estructuras construidas en sistema de pórticos?

Figura 1. Sistema estructural de pórticos.

1.2 JUSTIFICACIÓN

La Norma Colombiana para Construcción Sismo Resistente denominada NSR-10 en su Título A indica que "El diseño, construcción y supervisión técnica de edificaciones en el territorio de la República de Colombia debe someterse a los criterios y requisitos mínimos que se establecen en la Normas Sismo Resistentes Colombianas.[5]"

Una edificación diseñada siguiendo los requisitos de la NSR-10, debe ser capaz de resistir, además de las fuerzas que le impone su uso, temblores de poca intensidad y moderados sin daño estructural, pero posiblemente con algún daño a los elementos no estructurales y un temblor fuerte con daños a elementos estructurales y no estructurales pero sin colapso. [6]

Además de la defensa de la vida, con el cumplimiento de los niveles prescritos por el reglamento para los movimientos sísmicos de diseño se permite proteger en alguna medida el patrimonio de los colombianos.[7]

[5] NSR-10 objeto A.1.1.1, Título A - Requisitos generales de diseño y construcción sismo resistente, pág. 1
[6] NSR-10 objeto A.1.2.2.2, Título A - Requisitos generales de diseño y construcción sismo resistente, pág. 2
[7] NSR-10 objeto A.1.2.2.3, Título A - Requisitos generales de diseño y construcción sismo resistente, pág. 2

Basados en la problemática, se reconoce la necesidad que la ingeniería de telecomunicaciones sirva de apoyo y preste atención a la comunidad en general ayudando al cumplimiento de los objetos mencionados.

Por ende se pretende medir de forma sistemática la calidad del trabajo de las constructoras, realizando interventoría constante y automatizada a través de las telecomunicaciones.

2.OBJETIVOS

2.1 OBJETIVO GENERAL

Diseñar y probar un sistema domótico para medir los desplazamientos horizontales y verticales de una estructura en sistema de pórticos a escala con sensores en sus columnas; para que a través de las telecomunicaciones, la transmisión y el procesamiento de la información obtenida se modele un método de alerta y permita a futuro en un ambiente real reducir los riesgos de colapso o daños y salvaguardar vidas.

2.2 OBJETIVOS ESPECÍFICOS

- ✓ Establecer el tipo y la tecnología de sensores a utilizar en la medición de los desplazamientos horizontales y verticales de las estructuras.
- ✓ Definir la tecnología de transmisión implicada en el envío de la información.
- ✓ Seleccionar la plataforma que integrará los sensores y la tecnología de transmisión.
- ✓ Elaborar el algoritmo para la medición y transmisión de las variables físicas involucradas.
- ✓ Diseñar el prototipo que permita apoyar la NSR-10 pre y post construcción.
- ✓ Almacenar en un repositorio la información obtenida y procesarla para generar alertas, que permitan realizar un seguimiento por parte de los expertos de Ingeniera Civil en construcciones basadas en sistemas de pórticos.
- ✓ Realizar las pruebas del prototipo en el modelo a escala que permita verificar la fiabilidad del mismo.

3.MARCO REFERENCIAL

3.1 MARCO CONCEPTUAL TECNOLÓGICO

Para comprender la forma en que el prototipo funciona, a continuación se especifican algunos conceptos fundamentales.

3.1.1 Sistemas de medida

Se denomina sistema a la combinación de dos o más elementos, subconjuntos y partes necesarias para realizar una o varias funciones. En los sistemas de medida, esta función es la asignación objetiva y empírica de un número a una propiedad o cualidad de un objeto o evento, de tal forma que la describa. Es decir, el resultado de la medida debe ser: independiente del observador (objetiva), basada en la experimentación (empírica), y de tal forma que exista una correspondencia entre las relaciones numéricas y las relaciones entre las propiedades descritas.

Los objetivos de la medida pueden ser: la vigilancia o seguimiento de procesos, como es el caso de la medida de la temperatura ambiente, de los contadores de gas y de agua, de la monitorización clínica, etc.; el control de un proceso, como en el caso de un termostato o el control de nivel en un depósito; y también puede ser una necesidad de la ingeniería experimental.

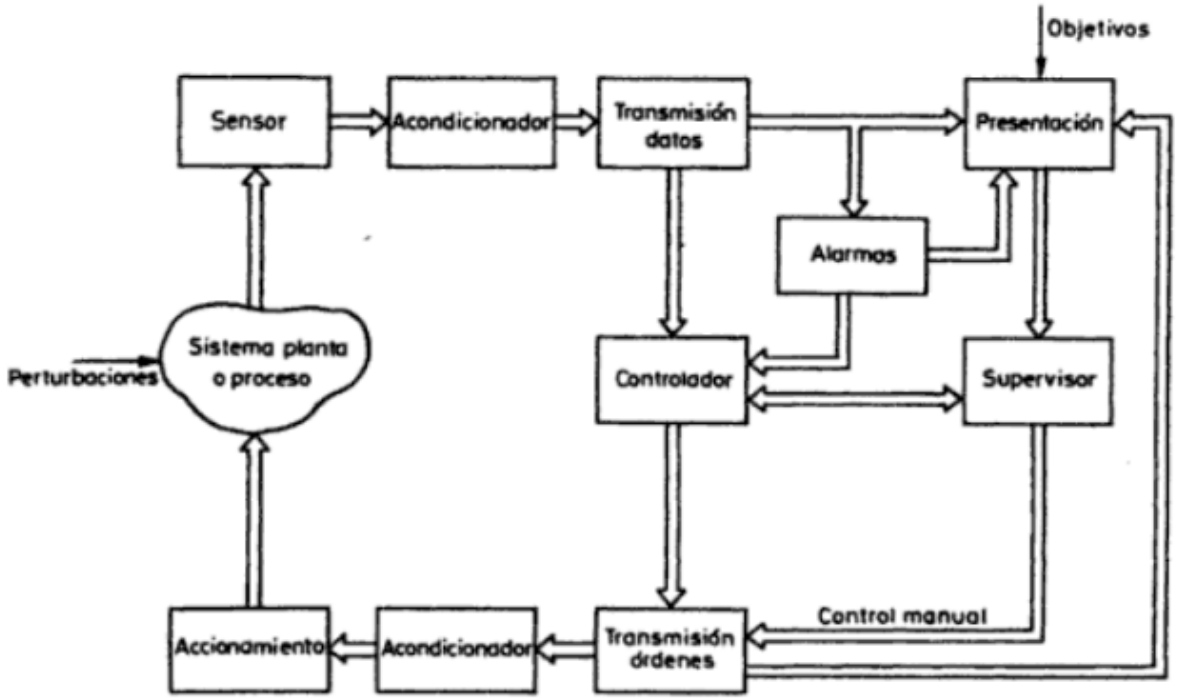
Las medidas en prototipos son además necesarias para verificar los resultados de los modelos desarrollados en un ordenador.

En la Figura 2 se describe la estructura general de un sistema de medida y control. En un sentido amplio, la realización de una medida implica, pues, además de la adquisición de la información, realizada por un elemento sensor o transductor, también el procesamiento de dicha información y la presentación de resultados, de forma que puedan ser percibidos por nuestros sentidos.

Cualquiera de estas funciones puede ser local o remota, implicando ello, en este segundo caso, la necesidad de transmitir la información.[8]

[8] RAMÓN, Areny. Sensores y acondicionadores de señal. Marcombo, 2004. p.1.

Figura 2. Estructura general de un sistema de medida y control[9]



3.1.2 Transductores, sensores y accionamientos

Se denomina transductor, en general, a todo dispositivo que convierte una señal de una forma física en una señal correspondiente pero de otra forma física distinta. Es, por tanto, un dispositivo que convierte un tipo de energía en otro. Esto significa que la señal de entrada es siempre una energía o potencia, pero al medir, una de las componentes de la señal suele ser tan pequeña que puede despreciarse, y se interpreta que se mide sólo la otra componente.

Dado que hay seis tipos de señales: mecánicas, térmicas, magnéticas, eléctricas, ópticas y moleculares (químicas), cualquier dispositivo que convierta una señal de un tipo en una señal de otro tipo debería considerarse un transductor, y la señal de salida podría ser de cualquier forma física “útil”.

Un sensor es un dispositivo que, a partir de la energía del medio donde se mide, da una señal de salida transductible que es función de la variable medida. Sensor y transductor se emplean a veces cómo sinónimos, pero sensor sugiere un significado más extenso: la ampliación de los sentidos para adquirir un conocimiento de cantidades físicas que, por su

[9] RAMÓN, Areny. Sensores y acondicionadores de señal. Marcombo, 2004. p.2.

naturaleza o tamaño, no pueden ser percibidas directamente por los sentidos. Transductor, en cambio, sugiere que la señal de entrada y la de salida no deben ser homogéneas. La tendencia es emplear el término sensor para designar el transductor de entrada, y el término actuador o accionamiento para designar el transductor de salida. Los primeros pretenden la obtención de información, mientras que los segundos buscan la conversión de energía.[10]

3.1.3 Materiales y efecto piezoeléctricos

El funcionamiento de los sensores que miden la aceleración de un punto de un cuerpo se fundamenta en la utilización del efecto piezoeléctrico, que se observan en varios cristales, como: el cuarzo, la turmalina, etc. Los acelerómetros que usan el cuarzo (óxido de silicio) en calidad de convertidor primario, adquirieron mayor utilización práctica inicialmente, que los que emplean otros cristales. Esto se debió a las notables cualidades del cuarzo tales como:

- No es higroscópico, es decir que no absorbe humedad
- Posee gran resistencia mecánica
- Presenta buenas cualidades aislantes
- Sus propiedades piezoeléctricas prácticamente no dependen de la temperatura, en un intervalo suficientemente amplio (20°C-40°C).

Sin embargo, su pequeña constante piezoeléctrica, comparada con los cristales artificiales actuales ha transformado su uso práctico, desplazándolo hacia el empleo como elemento sensible de acelerómetros patrones y de laboratorio.

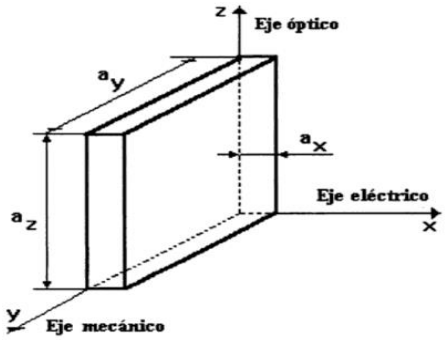
Si de un cristal de cuarzo, se corta una placa rectangular como la mostrada en la

[10] RAMÓN, Areny. Sensores y acondicionadores de señal. Marcombo, 2004. p.4.

Figura **3**, con las caras paralelas a los ejes (denominado corte de Curie), y la misma se somete a compresión o tracción a lo largo de su eje eléctrico, entonces, en las caras perpendiculares a dicho eje surgirán cargas electrostáticas de igual valor pero de signo contrario. Al pasar de la compresión a la tracción, los signos de las cargas cambian según la variación del signo de la fuerza que actúa a lo largo del eje eléctrico.[11]

[11] MOSQUERA, Genaro. Las vibraciones mecánicas y su aplicación al mantenimiento predictivo. Venezuela. p.129.

Figura 3. Corte de un cristal de cuarzo[12]



3.1.4 Micro controlador

Se llama microcontrolador a un sistema de microprocesador incluido todo él en un chip. Dentro de este chip están incluidos la CPU del procesador, memoria y elementos periféricos de forma que se pueda realizar todo un sistema de control simplemente conectando los elementos exteriores. Como todo sistema de procesador, el primer elemento necesario es la memoria de programa. Esta memoria de programa puede ser de tres tipos en los microcontroladores:

- Memoria *ROM*: El microcontrolador viene ya de fábrica con la memoria programada. Esto se realiza cuando se van a fabricar grandes series de equipos iguales; una vez realizado y probado suficientemente el prototipo, se encarga al fabricante del microcontrolador que fabrique la máscara del programa y los chips se suministran ya programados.
- Memoria *PROM*: La memoria de programa del microcontrolador viene sin programar de forma que el fabricante del equipo de control pueda efectuar la programación, pero una vez efectuada ésta ya no se puede modificar. Estas versiones de microcontrolador se llaman OTP (One Time Programming).
- Memoria *REPROM*: Una vez efectuada la programación de la memoria, ésta se puede borrar mediante rayos ultravioletas. Estas versiones son las que se utilizan para realizar los prototipos.

Existen también para la realización de prototipos versiones sin memoria de programa (ROMLESS) de forma que el programa se pueda escribir en una memoria externa bien del

[12] MOSQUERA, Genaro. Las vibraciones mecánicas y su aplicación al mantenimiento predictivo. p.130.

tipo *RAM* o del tipo *REPROM*. El segundo elemento incluido en el microcontrolador es la memoria de datos del tipo *RAM*.

Normalmente los microcontroladores suelen llevar de 1 Kbyte a 32 Kbyte de memoria de programa y de 128 bytes a 2 Kbytes de memoria de datos, pero en todos ellos suelen existir la posibilidad de ampliar estas memorias externamente.

El primer elemento periférico incluido en los microcontroladores son los puertos de entrada/salida. Normalmente se dispone de 2 a 6 puertos de entrada/salida de 8 bits que pueden ser programados individualmente, cada línea como entrada o como salida y con unas ciertas características. Estas líneas de entrada o salida al sistema se pueden utilizar también como líneas para unas ciertas funciones especiales de otros periféricos incluidos en el chip o como líneas de bus de direcciones y datos cuando se desea ampliar externamente la memoria comprendida en el chip.

El segundo elemento periférico que incluyen los microcontroladores son los temporizadores y contadores, que pueden funcionar o bien con el reloj interno del microprocesador incluyendo algún divisor de frecuencia o bien un reloj externo.[13]

3.1.5 Acelerómetros

Un acelerómetro es un dispositivo que mide la aceleración. La mayoría de los acelerómetros trabajan de manera indirecta. Llevan una cantidad de masa conocida, denominada masa sísmica, a una unión mecánica con el objeto que está siendo medido, de manera que cualquier aceleración que sufra el objeto medido, la masa sísmica debe experimentar la misma aceleración. Entonces el acelerómetro detectará la fuerza ejercida sobre la masa sísmica. El valor de la fuerza medida está relacionado con el valor de aceleración por la segunda ley de Newton: $a = \frac{F}{m}$

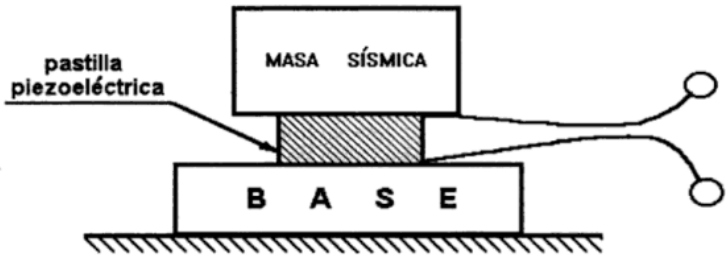
En dónde *F* es la medida por un transductor de fuerza y *m*, una cantidad conocida fija de masa. Por tanto, el transductor de fuerza puede estar calibrado para leer las unidades de aceleración.[14]

[13] SANTAMARÍA, Eduardo. Electrónica digital y microprocesadores. 1993. p. 257.
[14] MALONEY, Timothy J. Electrónica industrial moderna. Pearson, 2006. P 439.

3.1.5.1 Principio de operación

Un acelerómetro es un instrumento sísmico lineal que utiliza elementos piezoeléctricos de tal manera que se genera una carga electrostática proporcional a la aceleración aplicada al instrumento. Este sensor sísmico piezoeléctrico ideal se representa en la Figura 4.

Figura 4. Esquema de un acelerómetro ideal[15]



La cerámica poli cristalina o el cristal piezoeléctrico que produce la carga actúa como un muelle. Cuando se le aplica una aceleración a la base esta hace variar las tensiones en la pastilla dando lugar a la deformación de esta última.[16]

3.1.5.2 El acelerómetro piezoeléctrico

Los convertidores son elementos sensibles que transforman los cambios de la cantidad de una magnitud mecánica en variaciones de otra propiedad física, que suele ser, frecuentemente, una señal eléctrica proporcional al parámetro del movimiento mecánico desarrollado.[17] En la categoría de elementos sensibles con generación propia, los más utilizados son los de principio piezoeléctrico, electrodinámico y electromagnético.[18]

Montado el elemento sensible, que utiliza la conversión por generación propia y directa del principio piezoeléctrico en la estructura correspondiente, se convierte en el instrumento primario denominado acelerómetro, que goza de las ventajas generales siguientes: es ligero, robusto, posee amplia respuesta de frecuencia, tiene buena resistencia a relativamente altas temperaturas y costos de fabricación moderados.

Actualmente es el sensor más difundido a nivel mundial para la medición de variables mecánicas.

---

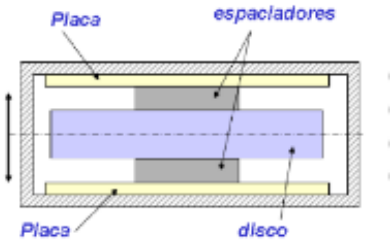
[15] MOSQUERA, Genaro. Las vibraciones mecánicas y su aplicación al mantenimiento predictivo. Venezuela. p.110.
[16] MOSQUERA, Genaro. Las vibraciones mecánicas y su aplicación al mantenimiento predictivo. Venezuela. p.110.
[17] MOSQUERA, Genaro. Las vibraciones mecánicas y su aplicación al mantenimiento predictivo. Venezuela. p.109.
[18] MOSQUERA, Genaro. Las vibraciones mecánicas y su aplicación al mantenimiento predictivo. Venezuela. p.110.

Su principio de funcionamiento se resume de la manera siguiente: cuando un acelerómetro está sometido a vibraciones, la masa ejerce una fuerza variable en el elemento sensible. La carga electrostática generada por el elemento primario es proporcional a la aceleración que da lugar a la fuerza, y ésta a su vez, al movimiento relativo de la masa, el cual finalmente, es proporcional a la aceleración de la base en determinado intervalo de frecuencias.[19]

3.1.5.3 Acelerómetro capacitivo

Este tipo de sensor, consiste en un disco rígido entre dos espaciadores elásticos que están situados entre medias de dos placas de condensador (Figura 5). El desplazamiento del disco en relación a las dos placas es una medida de la aceleración.[20]

Figura 5. Acelerómetro capacitivo[21]



3.1.6 Sensores acústicos (ultrasónicos)

Análogamente al procedimiento de ecosondeo, los sensores emiten impulsos ultrasónicos de una frecuencia de aprox. 40 kHz y detectan el tiempo que tardan en llegar los impulsos de eco reflejados por los obstáculos. La distancia que hay hasta el obstáculo más cercano se calcula a partir del tiempo de propagación del primer impulso de eco llegado y de la velocidad del sonido en el aire de aprox.340 m/s.[22]

3.1.6.1 Estructura

Se compone de un convertidor de ultrasonidos (membrana de aluminio en cuyo lado interior hay pegada una pastilla piezocerámica) y una placa de circuitos impresos con electrónica de emisión y evaluación.

[19] MOSQUERA, Genaro. Las vibraciones mecánicas y su aplicación al mantenimiento predictivo. Venezuela. p.111
[20] MIGUELANEZ, Jose. Electrónica básica para TMA-s. p.235.
[21] MIGUELANEZ, Jose. Electrónica básica para TMA-s. p.235.
[22] BOSCH. Los sensores en el automóvil. 2002. p. 26.

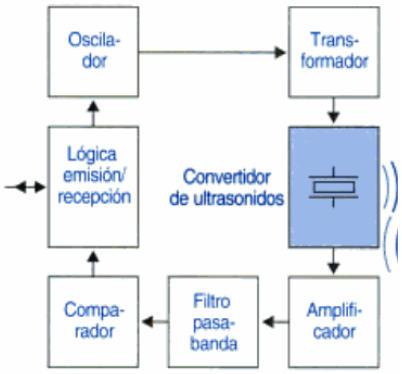
3.1.6.2 Funcionamiento

El sensor ultrasónico funciona según el principio “impulso-eco”.

En combinación con la triangulación. Cuando recibe de la unidad de control un impulso digital de emisión, el circuito electrónico excita la membrana de aluminio mediante impulsos rectangulares dentro de la frecuencia de resonancia para generar vibraciones típicas de 300 µs, emitiéndose entonces ondas ultrasónicas: la onda sonora reflejada por el obstáculo hace vibrar a su vez la membrana, que entretanto se había estabilizado (durante el período de extinción de aproximadamente 900 µs no es posible ninguna recepción).

La piezocerámica convierte estas vibraciones en una señal eléctrica analógica, que la electrónica del sensor amplifica y transforma en una señal digital (Figura 6). El sensor tiene prioridad frente a la unidad de control y, al detectar una señal de eco, conmuta la conexión de la señal a “bajo potencial” (<0,5V). Si se encuentra una señal de eco en la línea, no se puede procesar la señal de emisión. Cuando la tensión se vuelve inferior al umbral de conmutación de 1,5 V en la línea de señales, la unidad de control incita al sensor a que realice la emisión.

Figura 6. Esquema de conjunto del sensor ultrasónico.[23]



A fin de poder cubrir una zona lo más extensa posible, el ángulo de detección es grande en el plano horizontal. En el plano vertical, por el contrario, es necesario que el ángulo sea pequeño, para evitar reflexiones perturbadoras procedentes del suelo.[24]

[23] BOSCH. Los sensores en el automóvil. 2002. p. 37.

[24] BOSCH. Los sensores en el automóvil. 2002. p. 37.

3.1.7 Wi-Fi

Wi-Fi es una tecnología que permite que una gran variedad de equipos informáticos (ordenadores, impresoras, discos duros, cámaras, etc.) puedan interconectarse sin necesidad de utilizar cables. La aplicación principal que está teniendo Wi-Fi en la actualidad es la de permitir que varios ordenadores de casa o de la oficina puedan compartir el acceso a internet (de ADSL o cable). No obstante, esta tecnología permite crear una red entre los distintos equipos para compartir todos sus recursos. Lo que hace que Wi-Fi funcione es un equipo conocido como punto de acceso

Una de las principales ventajas de Wi-Fi es que utiliza el mismo protocolo que internet (protocolo TCP/IP). Este protocolo lo utilizan también las redes locales de cable, por lo que interconectar una red Wi-Fi con internet o con una red local cableada es bastante simple. Para que un equipo informático pueda comunicarse con el punto de acceso, es necesario que disponga de un equipamiento conocido como adaptador de red. Éste es una pequeña unidad que se comunica con el punto de acceso vía radio.[25]

3.1.8 El protocolo TCP/IP

El protocolo es el elemento que hace posible que los distintos ordenadores repartidos por el mundo y conectados a la red intercambien información. El protocolo que utiliza internet es el TCP/IP.

En dicho protocolo, a cada ordenador se le asigna una dirección o nombre (dirección IP), única para cada uno de ellos e identificativa en la red. En realidad al hablar de TCP/IP, estamos hablando de dos protocolos diferentes que se combinan para facilitar el control y la transferencia por Internet. El funcionamiento de este protocolo es muy sencillo.

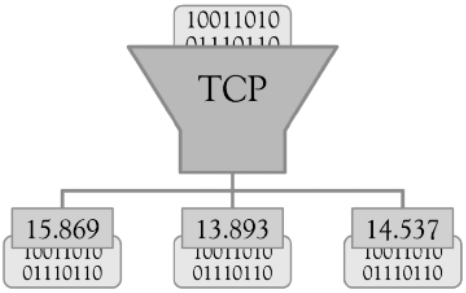
En primer lugar, el protocolo TCP fracciona (Figura 7), en paquetes independientes, la información y los numera para que al llegar a su destino se ordenen correctamente. Otro dato importante que incluye es la denominada suma de comprobación, que coincide con el número total de datos que contiene el paquete. Esta suma sirve para averiguar en el punto de destino si se ha producido alguna pérdida de información.[26]

---

[25] CARBALLAR, José Antonio. Wi-Fi : lo que se necesita conocer. Madrid. 2010. p. 1.

[26] RODRÍGUEZ, Ávila Abel. Iniciación a la Red de Internet. España. 2010. p. 7.

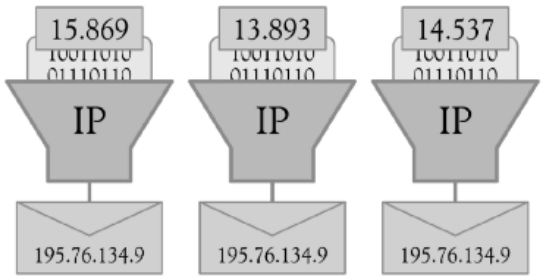
Figura 7. Fraccionamiento TCP[27]



A continuación, el protocolo IP proporcionará a cada paquete, entre otras informaciones, las direcciones IP de origen y destino (Figura 8).

Según son enviados, cada paquete irá escogiendo el camino más adecuado para llegar a su destino. De este modo internet se consolida como una red estable. Existen cientos de vías alternativas para un destino concreto, por lo que, aunque fallen ordenadores intermedios o no funcionen correctamente algunos canales de información, prácticamente los paquetes siempre llegarán a su destino.[28]

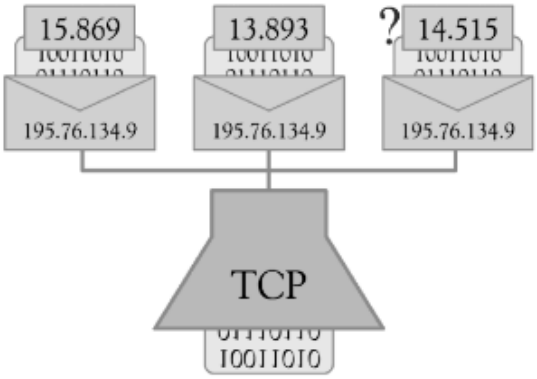
Figura 8. Asignación protocolo IP[29]



Cuando los paquetes lleguen a su destino vuelve a actuar el protocolo TCP, que realiza una nueva suma de comprobación y la compara con la suma original. Si alguna de ellas no coincide es síntoma de que se ha perdido información en el camino, por lo que solicita de nuevo el envío del paquete desde el origen (Figura 9). Por fin, cuando se ha comprobado la validez de todos los paquetes, el TCP los une formando el mensaje inicial.[30]

[27] RODRÍGUEZ, Ávila Abel. Iniciación a la Red de Internet. España. 2010. p. 7.
[28] RODRÍGUEZ, Ávila Abel. Iniciación a la Red de Internet. España. 2010. p. 8.
[29] RODRÍGUEZ, Ávila Abel. Iniciación a la Red de Internet. España. 2010. p. 8.
[30] RODRÍGUEZ, Ávila Abel. Iniciación a la Red de Internet. España. 2010. p. 8.

Figura 9. Comprobación y armado TCP del paquete[31]



3.2 MARCO CONCEPTUAL DE LAS VARIABLES FÍSICAS INVOLUCRADAS

3.2.1 Sistemas estructurales

Los sistemas estructurales de resistencia sísmica que reconoce la NSR-10 son los siguientes:

3.2.1.1 Sistema de pórticos

Es un sistema estructural compuesto por un pórtico espacial, resistente a momentos, esencialmente completo, sin diagonales, que resiste todas las cargas verticales y fuerzas horizontales (Figura 10).[32]

3.2.1.2 Sistema de muros de carga

Es un sistema estructural que no dispone de un pórtico esencialmente completo y en el cual las cargas verticales son resistidas por los muros de carga y las fuerzas horizontales son resistidas por muros estructurales o pórticos con diagonales.(Figura 10).[33]

3.2.1.3 Sistema combinado

Es un sistema estructural en el cual:

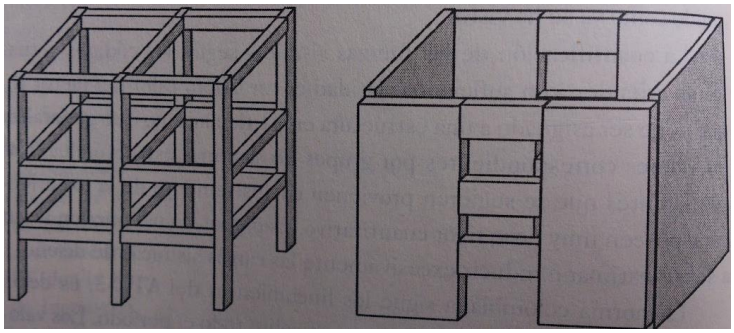
[31] RODRÍGUEZ, Ávila Abel. Iniciación a la Red de Internet. España. 2010. p. 8.

[32] AWAD, Roberto Rochel. Análisis y diseño sísmico de edificios.Medellín: Fondo editorial Universidad EAFIT, 2012, p. 63

[33] AWAD, Roberto Rochel. Análisis y diseño sísmico de edificios.Medellín: Fondo editorial Universidad EAFIT, 2012, p. 64

- Las cargas verticales son resistidas por un pórtico no resistente a momentos, esencialmente completo, y las fuerzas horizontales son resistidas por muros estructurales o pórticos con diagonales.
- Las cargas verticales y horizontales son resistidas por un pórtico resistente a momentos, esencialmente completo, combinado con muros estructurales o pórticos diagonales que no cumplen los requisitos de un sistema dual.[34]

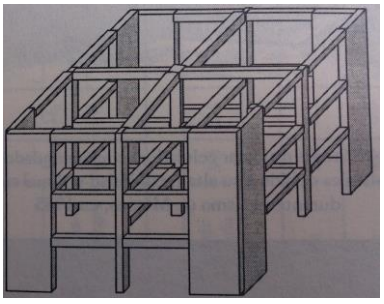
Figura 10. Sistemas estructurales de pórticos y de muros de carga



3.2.1.4 Sistema dual

Es un sistema estructural que tiene un pórtico espacial resistente a momentos y sin diagonales, combinado con muros estructurales o pórtico con diagonales. (Figura 11).[35]

Figura 11. Sistema dual



3.2.2 Evaluación de la deriva máxima

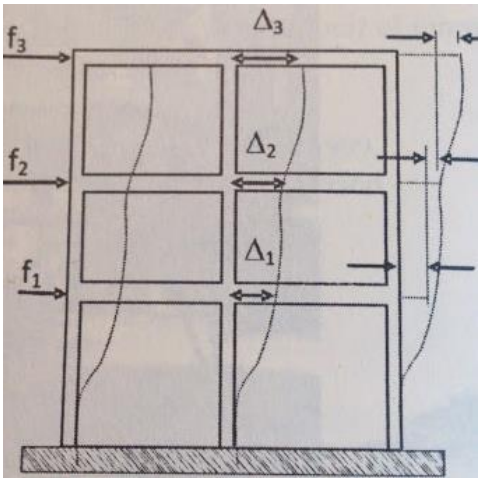
Las derivas son los desplazamientos relativos entre dos pisos consecutivos de una estructura, se evalúan para las cargas horizontales sin dividirlas por el factor de modificación de respuesta R. Al someter una estructura a la acción de las cargas

[34] AWAD, Roberto Rochel. Análisis y diseño sísmico de edificios. Medellín: Fondo editorial Universidad EAFIT, 2012, p. 64

[35] AWAD, Roberto Rochel. Análisis y diseño sísmico de edificios. Medellín: Fondo editorial Universidad EAFIT, 2012, p. 64

sísmicas, f_i, los entrepisos sufren desplazamientos horizontales, Δ_i los entrepisos sufren desplazamientos horizontales tal y como se muestra en la Figura 12. La magnitud de la deriva debe limitarse, entre mayor sea su valor, mayores serán los daños en los elementos no estructurales y en los acabados. Daños que son muy costosos de reparar y su magnitud puede ser tal que con posterioridad a un evento sísmico la estructura puede llegar a ser inhabitable.[36]

Figura 12. Derivas de entrepiso[37]



3.2.3 Asentamiento

Una de las funciones principales de los cimientos es transferir la carga total del edificio al suelo. La transferencia de la carga debe ser tan uniforme como sea posible, al mismo tiempo que debe extenderse sobre un área suficientemente grande para que sea segura. Por lo tanto el área que se precisa para la distribución de la carga depende de la capacidad del terreno para soportar la carga con un margen dado de seguridad. Esto tiene relación con un problema básico. El problema siguiente consiste en averiguar cómo se comporta el terreno una vez se le ha impuesto la carga.

En general, todos los suelos se compactan o endurecen, es decir, se produce un movimiento hacia abajo denominado asentamiento, al estar sometidos a una carga. La carga del suelo, aplicada a través de los cimientos, aumenta la presión del agua y del suelo. El agua se expulsa de entre las partículas sólidas y va a parar a zonas donde la presión es menor. Por otra parte, las partículas sólidas reducen su separación. La

[36] AWAD, Roberto Rochel. Análisis y diseño sísmico de edificios. Medellín: Fondo editorial Universidad EAFIT, 2012, p. 96.

[37] AWAD, Roberto Rochel. Análisis y diseño sísmico de edificios. Medellín: Fondo editorial Universidad EAFIT, 2012, p. 96.

consolidación continúa hasta que la presión del agua baja a su valor original y las fuerzas entre partículas aumentan hasta alcanzar un valor igual al de la carga que tengan aplicada. El asentamiento total dependerá del tipo de suelo y de la carga impuesta. También hay que tener en cuenta el factor tiempo. La carga aplicada al suelo aumenta a medida que la construcción avanza.

Debido a las variaciones naturales que se producen en todos los suelos, puede suceder que el asentamiento total no sea el mismo en todas partes. Este es un fenómeno que produce movimientos diferenciales y que ocurre aun cuando la carga del edificio no se ha distribuido uniformemente. Si, por otra parte, la carga del edificio no se ha distribuido uniformemente, aumentará el grado de asentamiento diferencial. Aunque debido a la naturaleza de los suelos suele ser imposible eliminar del todo las diferencias de los asentamientos, es siempre aconsejable asegurarse de que la carga del edificio se distribuye lo más uniformemente posible.[38]

3.2.4 Funciones trigonométricas

La trigonometría, cómo una rama propia de las matemáticas, tienen sus orígenes como herramienta para la elaboración de tablas astronómicas, y trata del uso de funciones trigonométricas para resolver problemas geométricos que implican triángulos. Es importante en diseño estructural y arquitectónico, astronomía y navegación, por ejemplo; en las ciencias físicas, las funciones trigonométricas son importantes para describir movimientos circulares y todo tipo de movimientos periódicos, incluido el movimiento ondulatorio.[39]

3.2.4.1 Definiciones geométricas

Las principales funciones trigonométricas del ángulo θ, el ángulo interno en A en la Figura 13, son el seno, el coseno y la tangente del ángulo.[40]

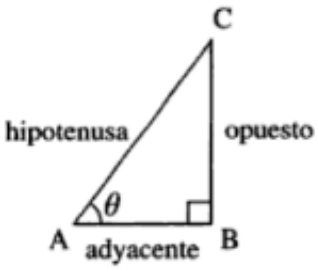
A partir de éstas funciones se definen otras funciones, las más importantes son la secante, la cosecante y la cotangente.

[38] ADDLESON, Lyall. Materiales para la construcción, Volumen 1. Reverte. 1983. p. 142.

[39] STEINER, Erich. Matemáticas para las ciencias aplicadas. Barcelona.2005. p. 49.

[40] STEINER, Erich. Matemáticas para las ciencias aplicadas. Barcelona.2005. p. 49.

Figura 13. Funciones trigonométricas[41]



$$\operatorname{sen}\theta = \frac{\text{opuesto}}{\text{hipotenusa}} = \frac{bc}{ac}$$

$$\cos\theta = \frac{\text{adyacente}}{\text{hipotenusa}} = \frac{ab}{ac}$$

$$\operatorname{tg}\theta = \frac{\text{opuesto}}{\text{adyacente}} = \frac{bc}{ab} = \frac{\operatorname{sen}\theta}{\cos\theta}$$

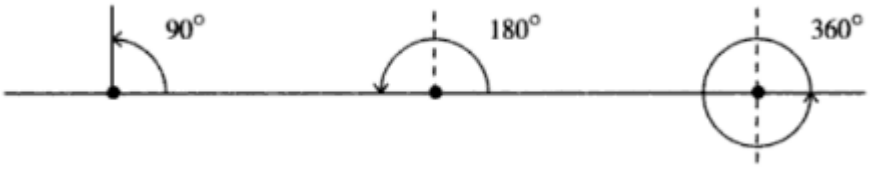
$$\sec\theta = \frac{1}{\cos\theta}, \qquad \operatorname{cosec}\theta = \frac{1}{\operatorname{sen}\theta}, \qquad \operatorname{cotg}\theta = \frac{1}{\operatorname{tg}\theta} = \frac{\cos\theta}{\operatorname{sen}\theta}.$$

3.2.5 Unidades de ángulo

La unidad de ángulo usual es el grado. La

Figura **14** muestra el ángulo recto (90°), el ángulo plano (180°) y el ángulo alrededor de un punto (360°).[42]

Figura 14. Ángulos[43]



3.3 MARCO DE ANTECEDENTES

Actualmente el campo de acción del monitoreo de estructuras es muy limitado, se logra identificar las siguientes empresas que prestan servicios orientados al campo previamente mencionado.

3.3.1 Servitopográficos

Empresa colombiana que provee “trabajo topográfico de control en el cual se cuantifica en el tiempo, el posible movimiento vertical de las construcciones, emplean equipos

[41] STEINER, Erich. Matemáticas para las ciencias aplicadas. Barcelona.2005. p. 50.
[42] STEINER, Erich. Matemáticas para las ciencias aplicadas. Barcelona.2005. p. 51.
[43] STEINER, Erich. Matemáticas para las ciencias aplicadas. Barcelona.2005. p. 51.

topográficos digitales para ello"[44], pero no a través de dispositivos dedicados para el manejo en tiempo real de las medidas implicadas.

3.3.2 Eurorva

Empresa Mexicana, que utiliza tecnología basada en el uso de fibra óptica para realizar un monitoreo continuo de una estructura con altos niveles de precisión[45].

3.3.3 Ambher Ingeniería

El monitoreo de edificios que ofrece Ambher Ingeniería tiene la finalidad de controlar la seguridad, comportamiento y el rendimiento del edificio a lo largo de su construcción y del tiempo. El monitoreo permanente permite la temprana detección de problemas estructurales como aquellos ocasionados por sismicidad, explosiones, vibraciones, torsiones y asentamientos[46].

3.3.4 Soluciones y Suministros para Ingenierías (SSI)

Empresa Colombiana de base tecnológica fundada en la ciudad de Santiago de Cali en el año 2006. Importa y comercializa equipos e instrumentos para el mercado de la ingeniería civil, electrónica, eléctrica, mecánica, de materiales, aeroespacial, de alimentos, químico y automotriz, entre otras.

Ofrece soluciones tecnológicas de desarrollo e integración a la medida, innovadoras y con la más alta calidad, para aplicaciones en geotecnia, geofísica, biotecnología, minas, petróleo, hidráulica, energía, ambiente, automatización, control de procesos, instrumentación y telemetría.

Suministra asesoría, soporte y capacitación de sistemas y equipos de sismología, monitoreo de salud estructural y ensayos de suelos en campo y laboratorio.[47]

---

[44] Servitopográficos, empresas [En línea]. < http://servitopograficos.com/index.php/empresa s> [2013]

[45] Eurorva, Servicio [En línea]. < http://www.eurorva.mx/monitoreo.html > [2011]

[46] Ambher Ingeniería, Sismiología [En línea]. <http://www.ambher.com/infraestructura?tag=Sismiolog%C3%ADa > [2014]

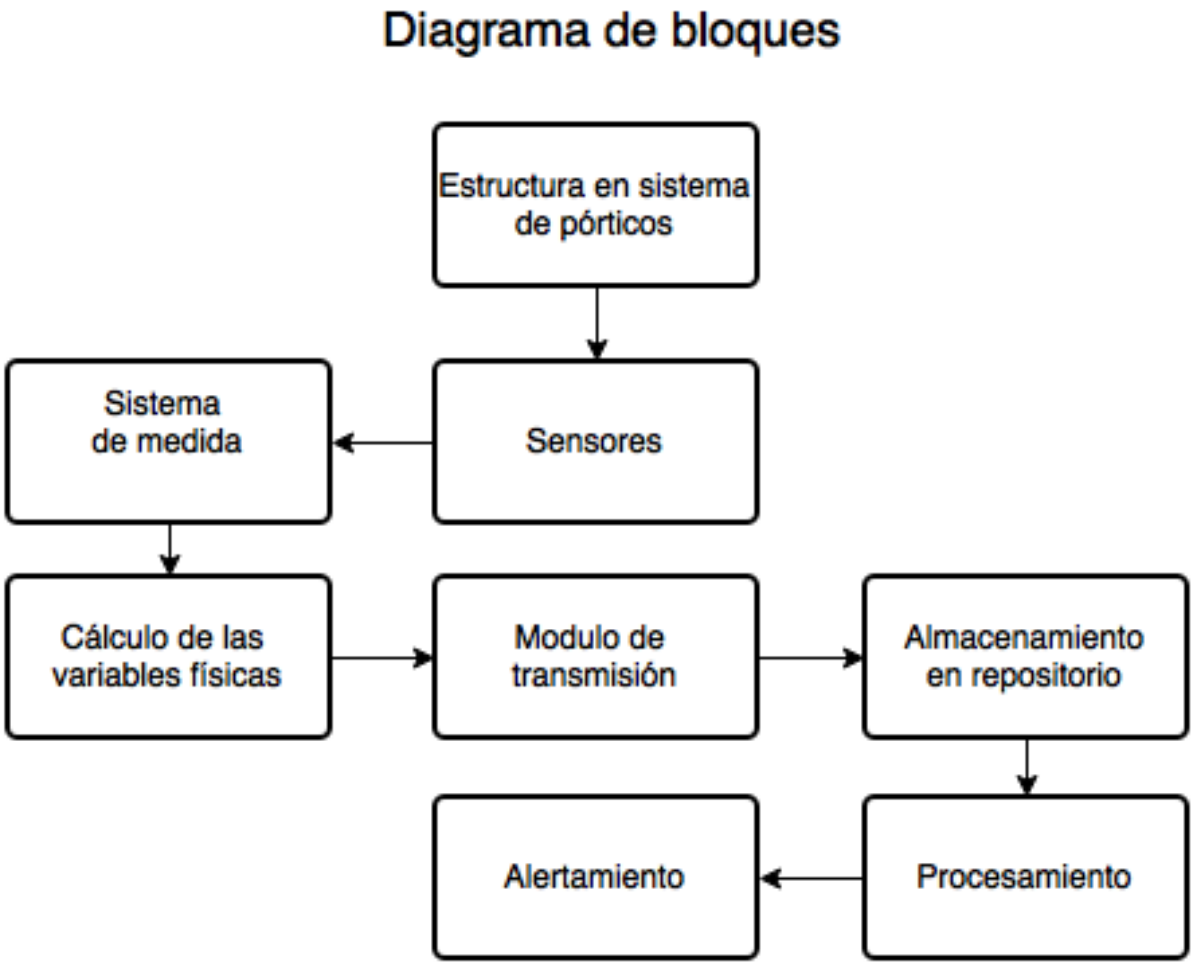
[47] SSI, [En línea]. < http://www.ssi.com.co/nosotros/>. [2015].

4.METODOLOGÍA

4.1 DIAGRAMA DE BLOQUES

Como parte del proceso de diseño, se pretende que el prototipo funcione como lo muestra el diagrama de bloques de la Figura 15.

Figura 15. Diagrama de bloques del prototipo.



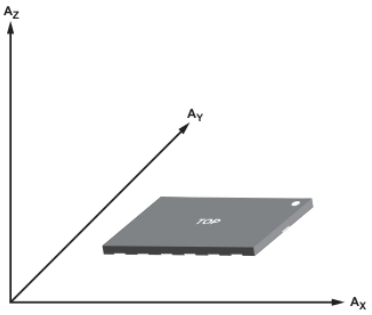
4.2 MÉTODOS DE MEDICIÓN DE LOS DESPLAZAMIENTOS.

Debido a la naturaleza del movimiento que se capturará, a continuación se muestra cómo se evaluarán los dos movimientos a medir (horizontales y verticales).

4.2.1 Medición de los movimientos horizontales (deriva).

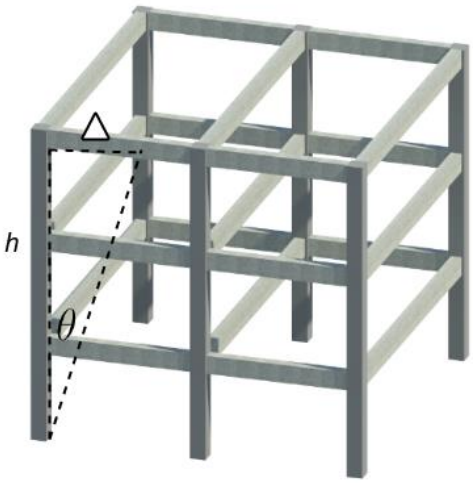
Teniendo en cuenta que la deriva en el sistema de pórticos debe ser una distancia máxima en la que la estructura se desplaza, se obtendrá mediante los sensores la inclinación en grados de la estructura en los ejes x e y del plano tridimensional, y en sus planos xy/yz. (Figura 16).

Figura 16. Ejes y planos tridimensionales[48]



Teniendo este valor en grados, y mediante las funciones trigonométricas de la Figura 13, procederemos a calcular el valor del desplazamiento gracias a la tangente de la siguiente forma, basados de igual manera en la los datos de la Figura 17:

Figura 17. Componentes del desplazamiento horizontal.



$$tan\theta = \frac{opuesto}{adyacente} \Longrightarrow tang\theta = \frac{\Delta}{h} \Longrightarrow \Delta = tang\theta * h$$

Se obtendrá el valor Δ multiplicando el resultado de la tangente de θ con el valor de la altura *h* de la columna del modelo a escala. El detalle de cómo se obtendrá dicho

[48] DFROBOT, ADXL345 datasheet. p. 22. 2016

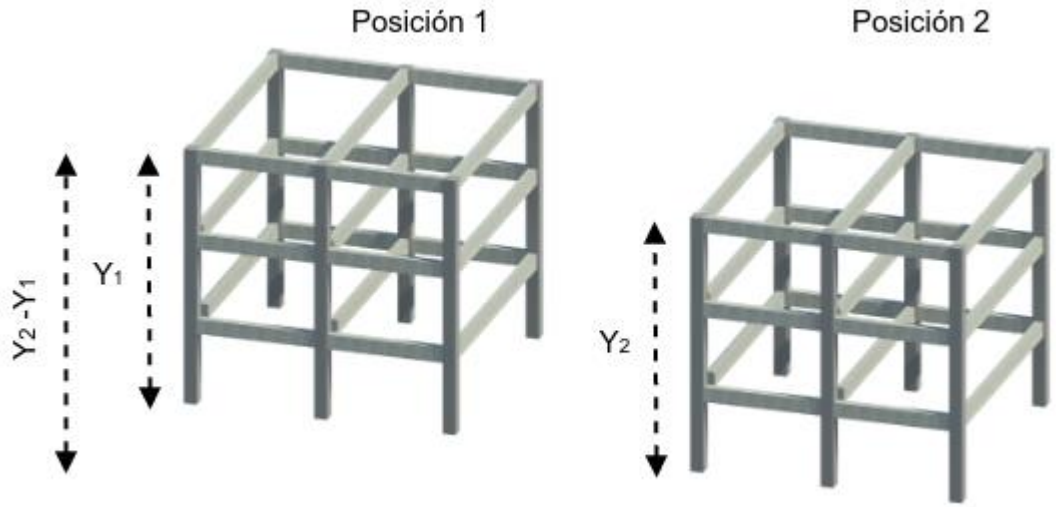
resultado a través de los sensores y la plataforma de integración se podrá apreciar en el apartado 6.2 “Almacenamiento y tratamiento de la información”.

4.2.2 Medición de los movimientos verticales (asentamiento).

El valor del desplazamiento vertical será estimado según la diferencia de distancias en que pueda moverse la estructura (Figura 18), en un sistema coordenado unidimensional:

“La distancia dirigida (dd) que existe de un punto P_1 a un P_2 viene dada por el valor final menos el inicial: $dd = P_2 - P_1$”[49]

Figura 18. Distancia dirigida para medir movimiento vertical



[49] UNAM. Matemáticas V... El Placer de Dominarlas Sin Complicaciones. México.2004. p. 73.

5.EVALUACIÓN TECNOLÓGICA

Basados en las descripciones de los movimientos que se van a medir, y las dos metodologías del apartado previo (ver página 38) que estarán implicadas en la adquisición de información, se ha realizado un estudio de los sensores que servirán para el desarrollo del prototipo, comparando las diferentes características y seleccionando uno para la etapa de implementación y experimentación. También se investigan las opciones para transmitir la información e integrar los sensores y dicha tecnología en una sola plataforma de operación.

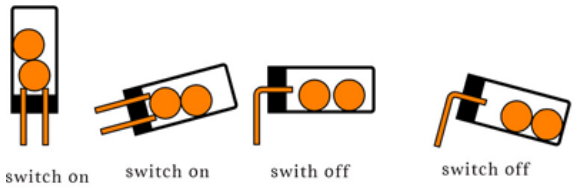
5.1 SENSORES

5.1.1 Sensores para el movimiento horizontal

5.1.1.1 Sensor de inclinación SW-520D

Un sensor de inclinación es un dispositivo que proporciona una señal digital en caso de que su inclinación supere un umbral. Se dispone de un cilindro cuya pared constituye un contacto eléctrico, mientras que el otro contacto está localizado en el centro de la base. Al inclinar lo suficiente el dispositivo ambas esferas constituyen un puente entre ambos contactos, cerrando el circuito (Figura 19). Este tipo de sensor no permite saber el grado de inclinación del dispositivo, simplemente actúa como un sensor que se cierra a partir de una cierta inclinación. Debido a su principio de funcionamiento, estos sensores resultan sensibles a movimientos bruscos y vibraciones.[50]

Figura 19. Modo operación sensor inclinación[51]



[50] Luis Llamas. Medir inclinación con arduino y sensor tilt sw-520d. [En línea]. <http://www.luisllamas.es/2015/08/medir-inclinacion-con-arduino-y-sensor-tilt-sw-520d/>. 2015.

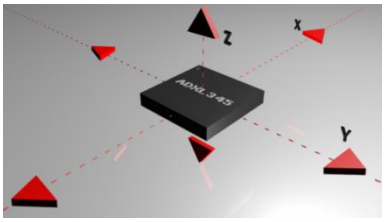
[51] Prometec. Sensor de inclinación. [En línea]. <http://www.prometec.net/tilt-switch/>. 2015.

5.1.1.2 Acelerómetro

Son dispositivos electromecánicos que detectan las fuerzas de aceleración, ya sea dinámica o estática, dentro de las cuales incluyen la gravedad para las estáticas, vibraciones y movimiento para las dinámicas.

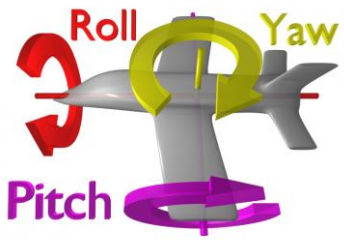
Los acelerómetros pueden medir la aceleración en uno, dos o tres ejes tal y cómo se muestra en la Figura **20**20. Los de tres ejes son más comunes debido a que los costos de producción de los mismos cada vez es más baja.

Figura 20. Acelerómetro de 3 ejes.[52]



En ausencia de una aceleración lineal, la salida del acelerómetro es el campo gravitacional de la tierra, con esta aceleración (gravedad) podemos calcular el ángulo de inclinación del sensor. Los ángulos de orientación/inclinación dependen del orden en el que se aplican las rotaciones. Comúnmente el orden utilizado es la secuencia aeroespacial de: derrape (*yaw*), rodar (*roll*), cabeceo (*pitch*) tal y como se muestra en la Figura 21. Con esta secuencia, se podría decir cómo gira un objeto por su eje X - Y a una postura determinada. Y a través de las coordenadas cartesianas y coordenadas polares (Figura 22) se llega a determinar la posición espacial.

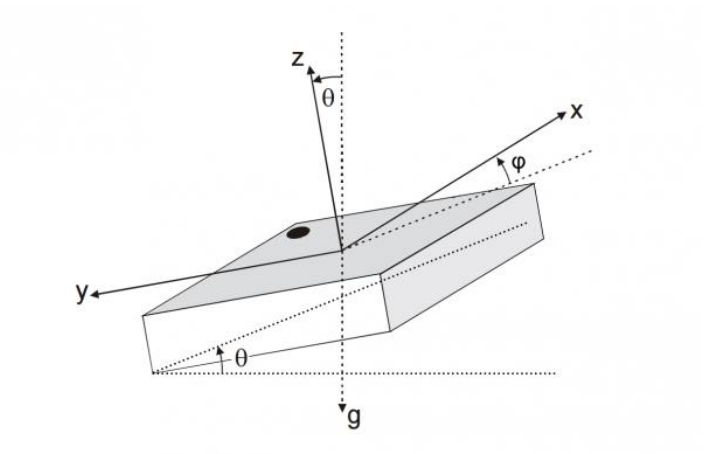
Figura 21. Componentes de la secuencia aeroespacial.[53]



[52] 5Hertz electrónica, ABC del acelerómetro. [En línea]. <http://5hertz.com/tutoriales/?p=228#4.2.1AC>

[53] DFROBOT, Tilt Sensing, [En línea]. <https://www.dfrobot.com/wiki/index.php?title=How_to_Use_a_Three-Axis_Accelerometer_for_Tilt_Sensing>

Figura 22. Coordenadas polares y cartesianas[54]



5.1.1.3 Tabla comparativa

Teniendo en cuenta el método del apartado 4.2.1, se realiza la siguiente tabla comparativa para determinar cuál de los estudiados es el mejor sensor para el prototipo.

Tabla 1. Sensores para el movimiento horizontal

Sensor	Sensibilidad	Medición en 3D	Costo
Sensor inclinación	Más de 10 °	No	$10 USD
Acelerómetro	Menos de 1 °	Si	$2,39 USD

5.1.1.4 Decisión técnica – Acelerómetro ADXL345

Para la realización del prototipo y la medición de los movimientos horizontales, se selecciona el acelerómetro debido a su capacidad de medir los ángulos de inclinación en los ejes X y Y, de igual forma, tiene mejores prestaciones dado que arroja un valor de dicho ángulo, en comparación con el sensor de inclinación, que simplemente después de superar cierta inclinación cerrará el circuito, adicional a que no posee la capacidad de medir en 3 dimensiones.

Otra bondad del acelerómetro, es que en el mercado existen muchas versiones de múltiples fabricantes, lo que ha llevado a la comunidad de experimentación a desarrollar

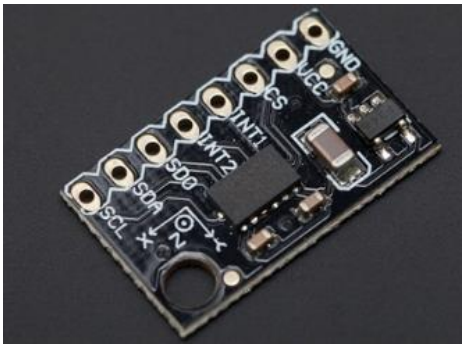
[54] DFROBOT, Tilt Sensing, [En línea]. <https://www.dfrobot.com/wiki/index.php?title=How_to_Use_a_Three-Axis_Accelerometer_for_Tilt_Sensing>

códigos y librerías para las plataformas de integración, lo que conlleva a que trabajar con el mismo sea más adecuado.

Se ha seleccionado el acelerómetro ADXL345 de DFROBOT (. El ADXL345 es un acelerómetro de 3 ejes pequeño, de baja potencia, delgado, con una medición de alta resolución (13 bits) en un máximo de ± 16 g. Datos de salida digital y es accesible a través ya sea de un SPI (3- o 4 hilos) o interfaz digital I2C.

El ADXL345 es muy adecuado para medidas de la aceleración de la gravedad estática en aplicaciones de detección de inclinación, así como de aceleración dinámica resultante del movimiento o shock. Su alta resolución (4 mg / LSB) permite la medición de cambios de inclinación **menores de 1,0 °.**[55]

Figura 23. Acelerómetro DFROBOT Adxl345[56]



Los detalles específicos de su forma de operación, valores de salida y demás se pueden apreciar en: **Anexo A. Datasheet acelerómetro ADXL345**

5.1.2 Sensores para el movimiento vertical

5.1.2.1 Sensor ultrasónico HC-SR04

Este sensor tiene un pequeño altavoz que emite una señal y un micrófono sensor que detecta la señal emitida. Internamente se calcula el tiempo que el sonido tarda en ir hasta un objeto y volver reflejado. A partir del tiempo que tarda el sonido y de la velocidad del sonido se puede calcular la distancia del sensor al objeto. El pitido (señal) emitido tiene una frecuencia de 40kHz. Esta frecuencia se encuentra muy por encima de 20kHz, que es

[55] DFROBOT, Triple Axis Accelerometer ADXL345, [En línea]. <http://www.dfrobot.com/index.php?route=product/product&product_id=383#.V4_GjXV97Zs>

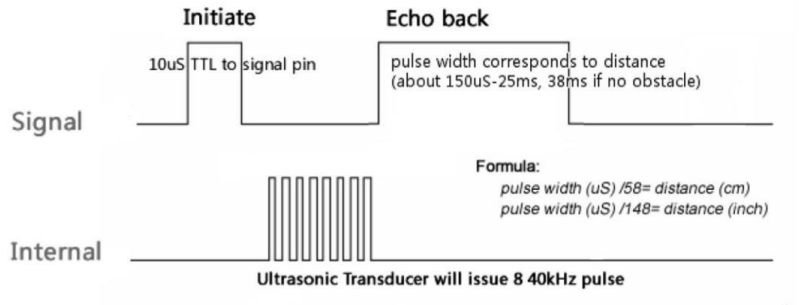
[56] DFROBOT, Triple Axis Accelerometer ADXL345, [En línea]. <http://www.dfrobot.com/index.php?route=product/product&product_id=383#.V4_GjXV97Zs>

la máxima frecuencia que los humanos pueden percibir. Por esta razón a este sonido de elevada frecuencia se le denomina ultrasonido.[57]

La Figura 25 muestra los pulsos recibidos y enviados por el sensor, de acuerdo a la hoja de datos.

Figura 24. Sensor ultrasónico HC-SR04[58]

Figura 25. Transductor ultrasónico[59]



Este sensor de ultrasonidos tiene las siguientes características:

- Distancia de detección: 2cm - 400cm
- Resolución: 0.3cm
- Frecuencia de sonido: 40kHz
- Ángulo eficaz: 15º
- Tensión de alimentación: 5V
- Consumo de corriente: 15mA

---

[57] Carlos Pardo. Sensor de distancia por ultrasonidos. [En línea]. <http://www.picuino.com/es/uno-doc/ultrasonic.html>

[58] Elecfreaks. Ultrasonic Ranging Module HC - SR04 datasheet.2015. p. 2.

[59] Geek Factory. Tutorial Sensor Ultrasónico HC-SR04. [En línea]. <http://www.geekfactory.mx/tutoriales/tutoriales-arduino/sensor-ultrasonico-hc-sr04-y-arduino/>

5.1.2.2 Sensor óptico Sharp GP2Y0A21YK0F

Es un sensor de medición de distancia (Figura 26), compuesto por una combinación integrada de PSD (*position sensitive detector*), IRED (infrared emitting diode) y un circuito de procesamiento de señales. La variedad de la reflectividad del objeto, la temperatura ambiental y la duración de funcionamiento no influencian fácilmente la detección de distancia debido a la adopción del método de triangulación. Este dispositivo da salida a la tensión correspondiente a la distancia de detección. Así que este sensor también se puede utilizar como un sensor de proximidad.[60]

Características:

- Distancia rango de medición: 10 a 80 cm
- Tipo de salida analógica
- Tamaño: 29.5 × 13 × 13,5 mm
- Consumo de corriente: 30 mA
- Alimentación: 4,5 a 5,5 V

Figura 26. Sensor SHARP GP2Y0A21YK0F[61]

5.1.2.3 Telémetro láser SF02/F

El módulo telémetro láser SF02/ F (Figura 27) proporciona mediciones de distancia rápidas y precisas en ambientes interiores y exteriores. Es ideal para su uso en vehículos aéreos no tripulados (UAV), aviones de radio control y robots.[62]

[60] SHARP Corporation, GP2Y0A21YK0F Datasheet. 2006. p. 1.

[61] SHARP Corporation, GP2Y0A21YK0F Datasheet. 2006. p. 1.

[62] LIGHTWARE Optoelectronics, drone-altimeters. [En línea]. <http://www.lightware.co.za/shop/en/drone-altimeters/7-sf02f.html>

Características:

- Peso 69 g
- Rango 0 - 50 m (objetivos naturales)
- Resolución: 1 cm
- Velocidad de actualización: 32 lecturas por segundo
- Precisión: ± (0,1% + 1) m
- Salidas e interfaces: analógica, serie y digital
- Voltaje de la fuente de alimentación: de 6,5 V - 9,0 V o 5,0 V ± 0,5 V DC
- Corriente: 150 mA (máxima)
- Potencia del láser: 10 W (pico típico), 10 mW (máxima media), Clase 1M
- Dimensiones : 27 x 59 x 86 mm
- Temperatura de funcionamiento: 0 - 40 ° C)
- Montaje: 4 x M3 (3,2 mm de diámetro)
- Divergencia del haz: 0.2 ° de material (típico)
- Material del lente: acrílico
- Conexiones: terminal de tornillo: 0,1 en la cabecera de campo

Figura 27. Telémetro láser SF02/F[63]

[63] LIGHTWARE Optoelectronics, drone-altimeters. [En línea]. <http://www.lightware.co.za/shop/en/drone-altimeters/7-sf02f.html>

5.1.2.4 Tabla comparativa

Teniendo en cuenta el método del apartado 4.2.2, se realiza la siguiente tabla comparativa para determinar cuál de los sensores estudiados es el mejor para el prototipo.

Tabla 2. Sensores para el movimiento vertical

Sensor	Rango mínimo	Rango máximo	Costo
Sensor ultrasónico HC-SR04	2 cm	400 cm	$ 3,39 USD
Sensor óptico Sharp	10 cm	80 cm	$ 13,59 USD
Telémetro láser SF02/F	1 cm	50 m	$ 299 USD

5.1.2.5 Decisión técnica – Sensor ultrasónico HC-SR04

Para la medición de los movimientos horizontales y el proceso de experimentación, se ha optado en primera instancia por el sensor ultrasónico HC-SR04 debido principalmente por su rango de medición, por su bajo costo y disponibilidad en el mercado.

Los detalles específicos, su forma de operación, valores de salida y demás se pueden apreciar en: **Anexo B. Datasheet sensor ultrasónico HC-SR04.**

5.2 TECNOLOGÍA DE TRANSMISIÓN

Teniendo en cuenta el despliegue que debería tener el/los prototipos, se opta por evaluar tecnologías de transmisión inalámbricas dado que son las menos intrusivas, más fáciles de desplegar y menos costosas, ya sea en un ambiente pre o post construcción. Teniendo en cuenta esto, se estimaría en un ambiente real tener dispositivos distribuidos de forma aparente tal y como se muestra en la

Figura **28**.

Figura 28. Sensores en estructura

Trasladándonos a un hipotético ambiente real en una edificación, es necesario contemplar tecnologías de transmisión de cobertura considerable, mayores de 10 metros para una estructura en sistema de pórticos, aunque dicho valor es relativo dependiendo de tamaño de la edificación.

Con ésta información se ha decidido evaluar entre las tecnologías de transmisión 3G/GSM y WiFi debido principalmente a su cobertura y popularidad actual.

5.2.1 Wi-Fi[64]

WiFi es una red de área local (LAN) que proporciona acceso a Internet dentro de un rango limitado. Utiliza frecuencias entre los 2,4 GHz y 5 GHz. Actualmente las normas de mayor auge para WiFi son la 802.11n y 802.11ac. Cada uno de estos nuevos estándares mejora el rendimiento en estas bandas de frecuencia. Su ancho de banda es alto, lo que significa que se puede consultar el correo electrónico y el flujo de vídeo y audio fácilmente de un teléfono, tableta o computadora si la intensidad de la señal es fuerte y si se está lo suficientemente cerca del punto de acceso.

Con cualquier tecnología inalámbrica, hay ventajas y desventajas. Lo que carece Wi-Fi en el rango, lo compensa con la velocidad y ancho de banda. Wi-Fi se puede utilizar para aplicaciones y despliegues de Internet de las cosas (IoT) que se ejecutan en un entorno local, o en un entorno distribuido como una "Area Wi-Fi extensa" como por ejemplo:

- Sistemas de seguridad en casa.
- Sensores de iluminación.
- Termostatos inteligentes en casa.

[64] Link-labs. Wifi-vs-cellular-differences-for-m2m. [En línea]. <http://www.link-labs.com/wifi-vs-cellular-differences-for-m2m/>

- Iluminación inteligente para las calles.
- Parquímetros.

Barcelona es gran ejemplo de ciudad inteligente con acceso Wi-Fi en toda la ciudad tal y cómo se muestra en la Figura 29.

Figura 29. Wi-Fi en la ciudad de Barcelona[65].

5.2.2 Red celular[66]

Es una red de área extensa (WAN) que cubre kilómetros. Se conecta a Internet a través de una serie de estaciones distribuidas en células por toda la ciudad. Cada una de estas estaciones base ofrecen cobertura de radio en un área amplia, y luego el total de la red de estaciones base puede dar acceso a través de todo el país, por ejemplo.

El ancho de banda es alto (se puede transmitir indicaciones del GPS, video y audio al igual que Wi-Fi), pero también depende de la cobertura. El acceso a los datos celulares, sin embargo, es más caro que el Wi-Fi y otros tipos de redes inalámbricas. El consumo de energía es alto, por lo que el punto final debe ser de fácil acceso.

[65] Ayuntamiento de Barcelona. Barcelonawifi. [En línea]. < http://www.bcn.cat/barcelonawifi/es/>. 2016.

[66] Link-labs. Wifi-vs-cellular-differences-for-m2m. [En línea]. <http://www.link-labs.com/wifi-vs-cellular-differences-for-m2m/>. 2014.

5.2.3 Tabla comparativa[67]

Tabla 3. Wi-Fi vs Red celular

Característica	Wi-Fi	Red celular
Tipo de red	Red de área local (LAN)	Red de área extensa (WAN)
Rango	100 m aprox.	Cualquier lugar con señal
Ancho de banda	Alto	Medio a alto
Tiempo de vida batería	7 días aprox.	1 -3 días aprox.
Costo del nodo cliente	$25 USD aprox.	$100 USD aprox.

De igual forma, se consulta en promedio el costo de un módulo Wi-Fi vs un módulo GSM/GPRS para una plataforma estándar de integración cómo Arduino teniendo los siguientes costos:

Tabla 4. Costo módulo para plataforma estándar

Wi-Fi	Red celular
Entre $6 y $25 USD aprox.	$340 USD aprox. Para 4G y $50 USD para 3G

5.2.4 Decisión técnica – Wi-Fi

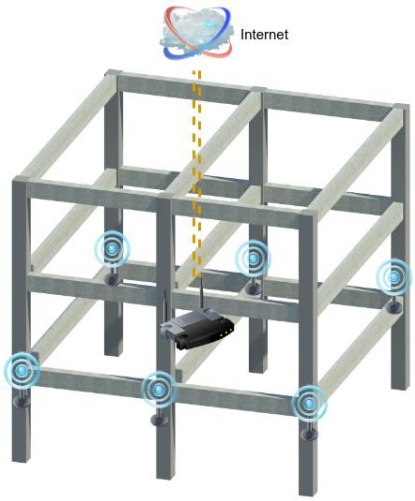
Debido a la relación costo beneficio, el conocimiento que se tiene acerca de la tecnología, se elige Wi-Fi cómo tecnología de transmisión de las variables físicas que se medirán. De igual forma no se requiere rangos de cobertura a gran escala para contemplar la red celular cómo tecnología de transmisión.

En ese orden de ideas, cada uno de los sensores deberán conectarse a un punto de acceso inalámbrico desde el cuál realizará el envío de la información y a su vez este estará conectado a Internet e las cosas, en la

[67] Link-labs. Wifi-vs-cellular-differences-for-m2m. [En línea]. <http://www.link-labs.com/wifi-vs-cellular-differences-for-m2m/>. 2014.

Figura **30** se muestra de forma trivial el diagrama de conectividad.

Figura 30. Diagrama conectividad Wi-Fi



Posterior a la elección de la plataforma de integración que se realizará en el apartado 5.3, se detallará en el mismo el tipo de módulo específico a utilizar (párrafo 5.3.5).

5.3 PLATAFORMA DE INTEGRACIÓN

Ya teniendo los tipos de sensores con los que se medirán las variables físicas y la tecnología de transmisión es momento de estudiar y elegir la arquitectura de integración de los anteriores.

Se ha optado por evaluar dos plataformas: Arduino y NodeMCU, la primera debido a su trayectoria, compatibilidad con sensores, variedad de modelos y conocimiento previo; la segunda cómo evaluación de una plataforma experimental y de reciente surgimiento.

5.3.1 NodeMCU

5.3.1.1 Definición y características

NodeMCU es un firmware basado en eLua para el módulo WiFi ESP8266 de Espressif[68]. El firmware NodeMCU es un proyecto que acompaña a los populares kits de desarrollo NodeMCU.[69]

[68] Espressif Systems (Shanghai) Pte. Ltd. es una compañía de semiconductores, con sede en Shanghai Zhangjiang parque de alta tecnología, que proporciona Wi-Fi de baja potencia y Bluetooth SoC, soluciones inalámbricas para aplicaciones de Internet de las cosas (IoT).

[69] NodeMCU, NodeMCU Documentation. [En línea]. https://nodemcu.readthedocs.io/en/master/

Elua significa Lua Embedded y el proyecto ofrece la plena aplicación del lenguaje de programación Lua incorporado al mundo, que se extiende con características específicas para el desarrollo de software embebido eficiente y portátil.[70]

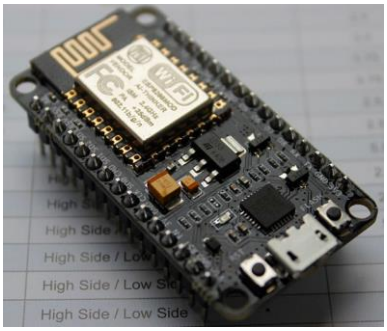
Lua es un lenguaje potente, eficiente, ligero e integrable con secuencias de comandos. Es compatible con la programación procedimental, la programación orientada a objetos, programación funcional, programación basada en datos, y descripción de datos. Lua se escribe de forma dinámica, lo que es ideal para la configuración, scripting, y prototipado rápido.[71]

En síntesis Nodemcu es una plataforma de código abierto, interactiva, programable, de bajo costo, simple, inteligente y habilitada con Wi-Fi.

5.3.1.2 Modelo NodeMCU DEVKIT V1.0

El kit de desarrollo para NodeMCU 1.0 es tal y cómo se muestra en la Figura 31. Con un cable micro USB se puede conectar NodeMCU DevKit a un computador para cargar los programas, al igual que Arduino.

Figura 31. NodeMCU



Es hardware abierto, con un núcleo ESP-12-E [32 Mbits (4 MBytes) versión de la flash].

El kit de desarrollo integra GPIO, PWM, 1-Wire y ADC todo en una placa, de los cuales:

- GPIO (*General Purpose Input/Output*, Entrada/Salida de Propósito General) es un pin genérico en un chip, cuyo comportamiento (incluyendo si es un pin de entrada o salida) se puede controlar (programar) por el usuario en tiempo de ejecución.[72]

[70] eLua Project. Overview. [En línea]. <http://www.eluaproject.net/overview>. 2011.
[71] Lua organization. About. En línea]. < http://www.lua.org/about.html>. 2016.
[72] Wikipedia, GPIO, [En línea]. <https://es.wikipedia.org/wiki/GPIO>. 2016.

- PWM (*pulse-width modulation,* modulación por ancho de pulsos) es una técnica de modulación de una señal o fuente de energía en la que se modifica el ciclo de trabajo de una señal periódica, una senoidal o una cuadrada, por ejemplo.[73]
- 1-Wire es un protocolo de comunicaciones en serie diseñado por Dallas Semiconductor. Está basado en un bus, un maestro y varios esclavos de una sola línea de datos en la que se alimentan.[74]
- ADC (Analog-to-Digital Conversion, conversión analógica-digital) Consiste en la transcripción de señales analógicas en señal digital, con el propósito de facilitar su procesamiento (codificación, compresión, etcétera) y hacer la señal resultante (digital) más inmune al ruido y otras interferencias a las que son más sensibles las señales analógicas.[75]

5.3.1.3 Diagrama esquemático de pines de entrada y salida

En la Figura 32 se logra ver la cantidad de pines de entrada y salida que posee el dispositivo NodeMCU.

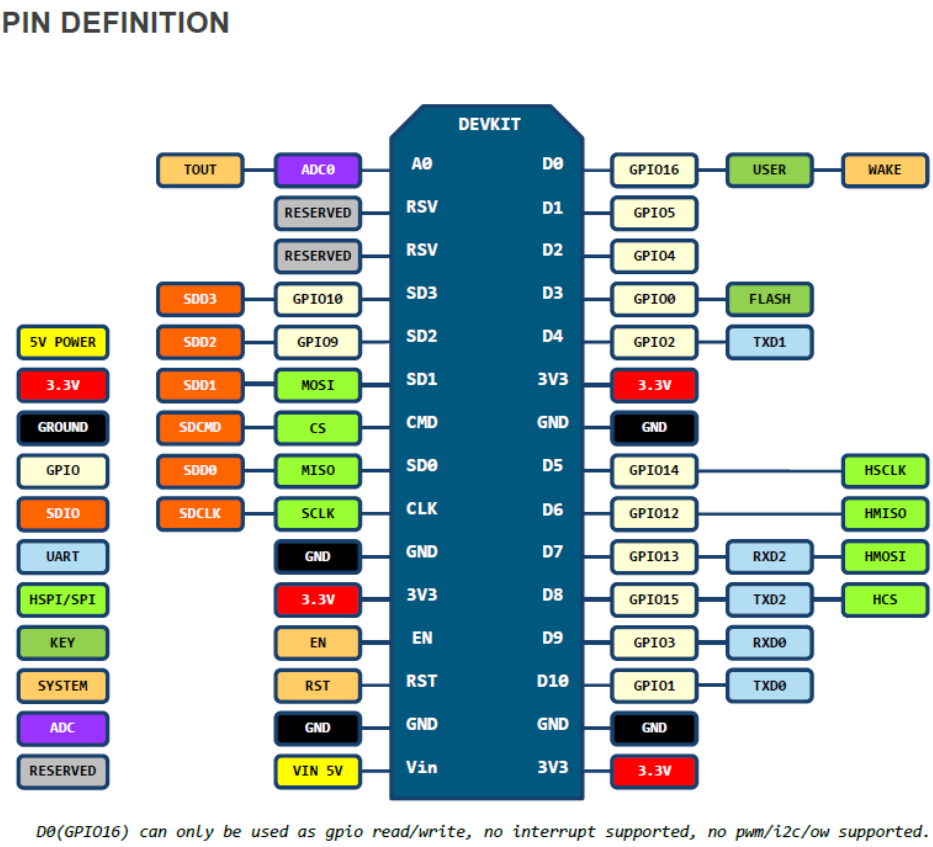
5.3.1.4 Observaciones adicionales

- La alimentación externa del dispositivo debe ser de 5V a través del pin *VIn*, lo que implica para el desarrollo del prototipo una fuente de alimentación que otorgue dicho voltaje, o en su defecto un circuito regulador de voltaje para usarlo con una batería estándar de 9V, lo que conlleva a la adquisición de hardware adicional y aumento en el "tamaño total" del prototipo.
- Se experimentó en primera medida la integración del acelerómetro a la plataforma, con resultados no satisfactorios. No existe un driver específico oficial del fabricante del acelerómetro ADXL345 de DFROBOT para que trabaje con NodeMCU y su lenguaje Lua. Se encuentra en la documentación oficial de un módulo para el modelo del acelerómetro (https://nodemcu.readthedocs.io/en/dev/en/modules/adxl345/) pero de otro fabricante (SparkFun). De igual forma no muestra la forma específica de traer los valores de inclinación requeridos para la metodología de medición del apartado 4.2.1.

[73] Wikipedia, PWM, [En línea]. <https://es.wikipedia.org/wiki/Modulaci%C3%B3n_por_ancho_de_pulsos>. 2016.
[74] Wikipedia, 1 Wire, [En línea]. <https://es.wikipedia.org/wiki/1-Wire>. 2016.
[75] Wikipedia, ADC, [En línea]. <https://es.wikipedia.org/wiki/Conversión_analógica-digital>. 2016.

- No existe un driver específico oficial del fabricante del ultrasonido HC-SR04 para que trabaje con el NodeMCU y su lenguaje Lua. Existe documentación de un desarrollador tercero (https://github.com/sza2/node_hcsr04) para trabajar con dicho sensor. De igual forma y teniendo en cuenta el datasheet para este sensor, el NodeMCU no posee pines con salida de voltaje a 5V, entrega voltaje a 3.3V. Sería necesario la adquisición de hardware o fuente adicional.
- Ya viene integrado con Wi-Fi, el proceso de conexión a través de Lua es simple.
- Lua es un lenguaje de programación relativamente reciente (1993), no se posee experiencia de programación de dicho lenguaje.

Figura 32. Distribución de pines NodeMCU[76]



[76] Github. nodemcu-devkit-v1.0. [En línea]. <https://github.com/nodemcu/nodemcu-devkit-v1.0>. 2016

5.3.2 Arduino

5.3.2.1 Definición y características

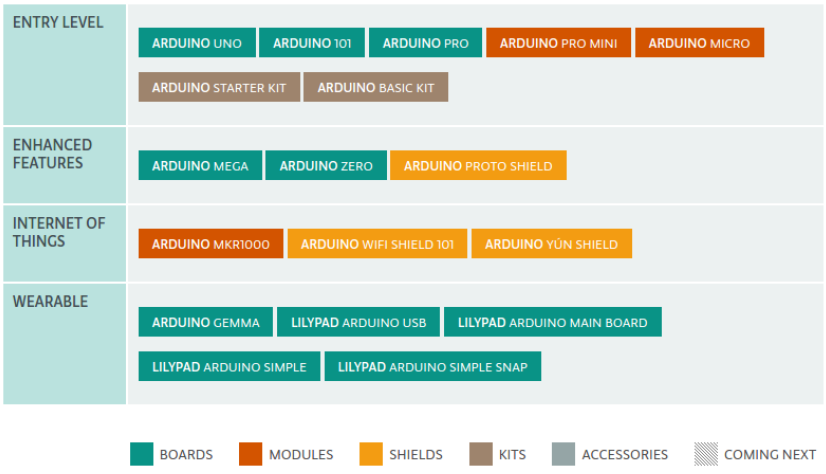
Arduino forma parte del concepto de hardware y software libre y está abierto para uso y contribución de toda la sociedad. Arduino es una plataforma de prototipos electrónicos, creado en Italia, que consiste básicamente en una placa microcontrolador, con un lenguaje de programación en un entorno de desarrollo que soporta la entrada y salida de datos y señales. Arduino es flexible y no requiere de un profundo conocimiento sobre el campo de la electrónica, lo que hizo que fuera muy popular entre los artistas y principiantes, además de los desarrolladores experimentados. Arduino es una plataforma de computación física (son sistemas que perciben la realidad y responden con acciones físicas), basada en una simple placa microcontrolador de entrada/salida y desarrollada sobre una biblioteca que simplifica la escritura de la programación en C/C++.

Arduino está basado en un microcontrolador (Atmega) y de esa forma se puede programar lógicamente, es decir, es posible la creación de programas, utilizando un lenguaje propio basado en C/C++, que, cuando se implementa hacen que el hardware ejecute ciertas acciones.[77]

5.3.2.2 Modelos

En la Figura 33 se visualiza los distintos modelos de la plataforma Arduino.

Figura 33. Variedad de productos Arduino[78]



[77] CAICEDO, Antonio. Arduino para Principiantes. 2014. p. 7.

[78] Arduino.cc. Arduino Products. [En línea]. < https://www.arduino.cc/en/Main/Products>. 2016.

Adicional a la cantidad de modelos que indicaron previamente, en la Tabla 5 es posible evaluar la comparación de las características principales de todos los modelos vigentes de Arduino.

Tabla 5. Especificaciones modelos Arduino[79]

Name	Processor	Operating/Input Voltage	CPU Speed	Analog In/Out	Digital IO/PWM	EEPROM [kB]	SRAM [kB]	Flash [kB]	USB	UART
101	Intel® Curie	3.3 V/ 7-12V	32MHz	6/0	14/4	-	24	196	Regular	-
Gemma	ATtiny85	3.3 V / 4-16 V	8 MHz	1/0	3/2	0.5	0.5	8	Micro	0
LilyPad	ATmega168V ATmega328P	2.7-5.5 V / 2.7-5.5 V	8MHz	6/0	14/6	0.512	1	16	-	-
LilyPad SimpleSnap	ATmega328P	2.7-5.5 V / 2.7-5.5 V	8 MHz	4/0	9/4	1	2	32	-	-
LilyPad USB	ATmega32U4	3.3 V / 3.8-5 V	8 MHz	4/0	9/4	1	2.5	32	Micro	-
Mega 2560	ATmega2560	5 V / 7-12 V	16 MHz	16/0	54/15	4	8	256	Regular	4
Micro	ATmega32U4	5 V / 7-12 V	16 MHz	12/0	20/7	1	2.5	32	Micro	1
MKR1000	SAMD21 Cortex-M0+	3.3 V/ 5V	48MHz	7/1	8/4	-	32	256	Micro	1
Pro	ATmega168 ATmega328P	3.3 V / 3.35-12 V 5 V / 5-12 V	8 MHz 16 MHz	6/0	14/6	0.512 1	1 2	16 32	-	1
Pro Mini	ATmega328P	3.3 V / 3.35-12 V 5 V / 5-12 V	8 MHz 16 MHz	6/0	14/6	1	1	32	-	1
Uno	ATmega328P	5 V / 7-12 V	16 MHz	6/0	14/6	1	2	32	Regular	1
Zero	ATSAMD21G18	3.3 V / 7-12 V	48 MHz	6/1	14/10	-	32	256	2 Micro	2

5.3.2.3 Pines entrada y salida

Los pines de entrada y salida y demás características es posible verificarlas en el:**Anexo C. Arduino UNO R3 pinout Diagram**

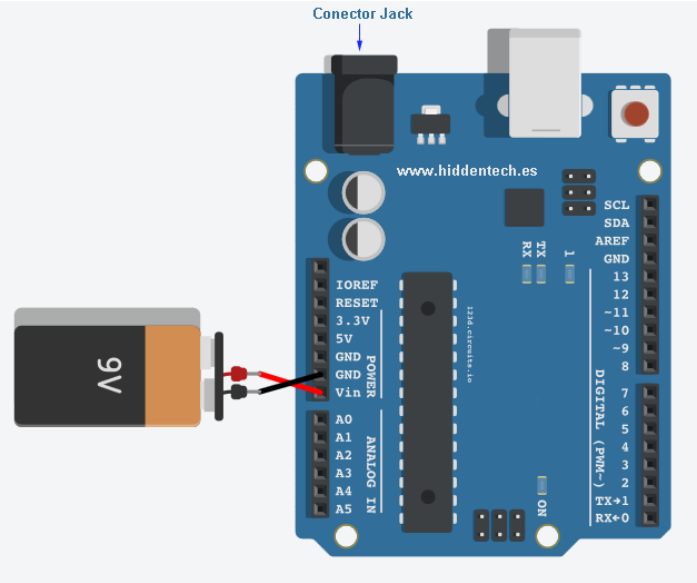
5.3.2.4 Observaciones adicionales

- Arduino permite la alimentación externa mediante una batería externa de 9V o un adaptador de corriente que esté entre los 9 y 12V, ya posee el conector integrado

[79] Arduino.cc. Arduino Products compare. [En línea]. <https://www.arduino.cc/en/Products/Compare>.2016.

en la plataforma, también es posible alimentarlo mediante el pin *Vin* según se puede ver en la Figura 34.

Figura 34. Alimentación externa Arduino



- Se tiene completa compatibilidad con el acelerómetro ADXL345 mediante librerías tales como “Wire.h”[80] para la lectura de las variables físicas.
- NewPing[81] es la librería para Arduino dedicada para el manejo de sensores ultrasónicos tales como el HCSR04.
- La alimentación de los dos sensores está garantizada debido a que la plataforma Arduino maneja salidas de alimentación a 3.3 y 5V.
- Arduino es modular, permite la adición de dispositivos Wi-Fi tales como el Wi-Fi shield.
- El lenguaje de programación es sencillo y ha sufrido su respectivo proceso de maduración, adicional, la comunidad de Arduino es enorme, se tiene gran cantidad de foros, ejemplos y guías para el desarrollo de aplicaciones.

[80] https://www.arduino.cc/en/Reference/Wire
[81] http://playground.arduino.cc/Code/NewPing

5.3.3 Tabla comparativa

Teniendo en cuenta las dos plataformas evaluadas, mediante la Tabla 6 se comparan las mismas teniendo en cuenta los sensores elegidos para la medida de las variables físicas y su completa compatibilidad, conexión, alimentación, etc.

Con dicha información se tomará la decisión acerca de la plataforma que integrará los sensores y la tecnología de transmisión.

Tabla 6. Tabla comparativa plataforma de integración

Plataforma	Alimentación externa	Voltaje correcto para los sensores	Nivel de complejidad programación	Librerías disponibles	Wi-Fi embebido	Costo
NodeMCU	Si, sólo a 5V	No, se requiere evaluar alimentación externa	Alta, no se tiene experiencia con el código.	No	Si	$ 16 USD
Arduino	Si, entre 9 a 12V	Si, entrega alimentación correcta.	Media, la lógica de programación es simple.	Si	No, se debe adicionar el módulo.	$ 14 USD

5.3.4 Decisión técnica – Arduino como plataforma de integración

Debido a su amplia gama de modelos, forma de alimentación externa sin hardware o electrónica adicional, su opción doble en voltaje de salida (3.3 y 5V), su simplicidad a la hora de realizar la programación mediante el software multiplataforma gratuito denominado Arduino IDE, su enorme comunidad de desarrolladores/experimentadores, múltiples foros y guías, librerías específicas para los sensores ADXL345 y HC-SR04 y su relación costo beneficio se opta por elegir Arduino UNO R3 (

) como plataforma de integración de las tecnologías antes seleccionadas.

La única desventaja encontrada es la adición de hardware para la transmisión de la información a través de Wi-Fi, la cual será evaluada y elegida en el apartado 5.3.5 del documento.

Las especificaciones técnicas del Arduino UNO R3 se encuentran en la Tabla 7.

Figura 35. Arduino UNO R3[82]

Tabla 7. Especificaciones técnicas Arduino UNO R3[83]

Característica / Componente	Valor / Referencia
Microcontrolador	ATmega328P
Voltaje de operación	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (límite)	6-20V
Pines I/O digitales	14 (de los cuáles 6 proveen salida PWM)
Pines PWM I/O digitales	6
Pines entrada analógicos	6
Memoria Flash	32 KB (ATmega328P)
Velocidad de reloj	16 MHz

Los detalles técnicos acerca de la plataforma de integración seleccionada pueden ser observados en:**Anexo D. Datasheet Arduino UNO R3**

5.3.5 Módulo Wi-Fi para la plataforma de integración elegida

Debido a la tecnología de transmisión seleccionada y a que la plataforma de integración no posee un módulo embebido de la misma, se indaga y se comparan dos tipos de módulos WiFi.

5.3.5.1 Arduino Wi-Fi shield

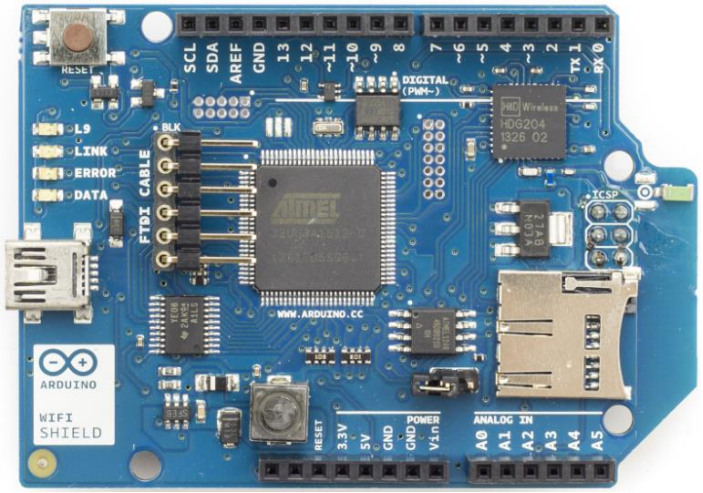
El Wi-Fi shield para Arduino (Figura 36) permite a la placa conectarse a Internet a través de la especificación inalámbrica 802.11 (Wi-Fi). Se basa en el chip LAN HDG204 802.11b/g de red inalámbrica en un sólo paquete. Proporciona red con el protocolo IP para manejo de TCP y UDP. Posee una librería dedicada para la elaboración del código.

[82] Arduino. Store. [En línea]. <https://store.arduino.cc/product/GBX00066>. 2016.

[83] Arduino. ArduinoBoardUno. [En línea]. <https://www.arduino.cc/en/Main/ArduinoBoardUno>. 2016.

Posee una ranura para tarjetas micro-SD, que se puede utilizar para almacenar archivos o como servidor través de la red. Es compatible con el Arduino Uno y Mega.

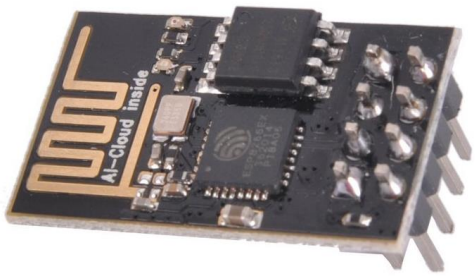
Figura 36. WiFi shield[84]



5.3.5.2 ESP8266

El módulo ESP8266 de Espressif (Figura 37) ofrece una solución altamente integrada Wi-Fi SoC (*System on a Chip*) para satisfacer las demandas continuas para el uso eficiente de la energía, el diseño compacto y un rendimiento fiable en la industria del IoT (Internet de la cosas).

Figura 37. Módulo ESP8266



Integra conmutadores de antena, RF balun, amplificador de potencia, receptor/ amplificador de bajo ruido, filtros y módulos de administración de energía. El diseño

[84] Arduino. ArduinoWiFiShield. [En línea]. <https://www.arduino.cc/en/Main/ArduinoWiFiShield>. 2016.

compacto reduce al mínimo el tamaño del circuito impreso adicional a que requiere poca circuitería adicional para trabajar.[85]

El módulo cuenta con las siguientes especificaciones técnicas principales:

- Protocolos Wi-Fi 802.11 b/g/n/e/i
- Rango de frecuencias: 2.4 G ~2.5 G (2400M ~ 2483.5M)
- Voltaje de operación 3.0 V ~ 3.6 V
- Modos Wi-Fi: estación/softAP/SoftAP+estación
- Seguridad WPA/WPA2
- Protocolos de red: Pv4, TCP/UDP/HTTP/FTP
- Configurable por comandos AT

5.3.5.3 Tabla comparativa

Teniendo en cuenta los modelos, se realiza la teniendo en cuenta primordialmente el tamaño y el costo de los módulos WiFi, debido a que el estándar ratifica que soporta la comunicación y los dos módulos realizan prácticamente lo mismo.

Tabla 8. Tabla comparativa módulos WiFi

Módulo	Protocolos compatibles	Tamaño/ peso	Costo
WiFi shield	802.11b/g	6,86 X 5.34 cm / 25 g	$100 USD
ESP8266	802.11 b/g/n/e/i	2.54 x 1.02 x 1.27 cm / 5,6 g	$ 7.45 USD

5.3.5.4 Decisión técnica – ESP8266

Se opta por usar el módulo ESP8266 debido a su bajo costo, la cantidad de protocolos compatibles. Adicional por su tamaño/peso no es intrusivo con la plataforma de integración y permite tener bajo peso en el prototipo final.

Los detalles técnicos acerca del módulo WIFi ESP8266 seleccionado pueden ser observados en:**Anexo E. Datasheet ESP8266**

[85] Espressif. ESP8266EX Datasheet. Overview. 2016. p.1.

6.EXPERIMENTACIÓN Y DESARROLLO DEL PROTOTIPO

En este apartado se mostrará la forma en que se integraron los sensores y el módulo de transmisión a la plataforma Arduino, mostrando los diferentes pasos realizados, las conexiones físicas necesarias para la correcta operación, así como el código del microcontrolador. De igual forma, se tratará en detalle la forma en que se manejará la información obtenida a través de una plataforma del internet de las cosas.

6.1 INTEGRACIÓN SENSORES Y MÓDULO WIFI CON LA PLATAFORMA ARDUINO

6.1.1 Integración acelerómetro ADXL345

6.1.1.1 Especificaciones generales a considerar del ADXL345

Según la hoja de datos del Acelerómetro, las siguientes consideraciones generales se tendrán en cuenta para la conexión con el Arduino UNO:

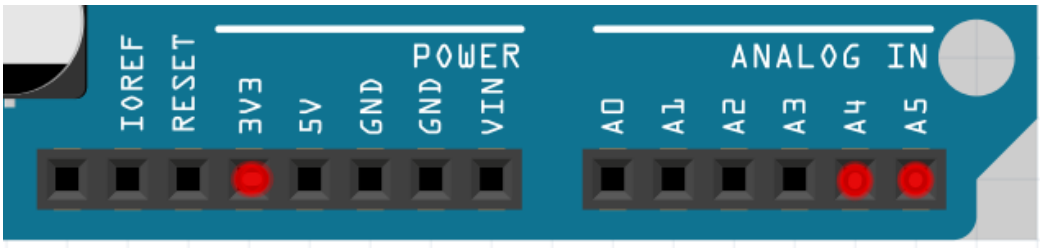
- Voltaje de funcionamiento: 3.3 ~ 6V
- Interfaz de comunicación I2C / SPI

El bus I2C fue diseñado por Philips a principios de los años 80 para permitir una fácil comunicación entre los componentes que se encuentran en la misma placa de circuito.[86]

6.1.1.2 Pines a utilizar según especificaciones

La plataforma Arduino es completamente compatible con dispositivos I2C, en la tarjetas Arduino con la etiqueta R3, el SDA (*Data line*) y el SCL (*clock line*) están en los pines cercanos al pin AREF. Específicamente para el Arduino Uno R3 corresponde a los pines de entrada análoga A4 (SDA), A5 (SCL) según cómo se muestra en la Figura 38.

Figura 38. Pines A4 (SDA), A5 (SCL) y alimentación 3.3V - Arduino UNO.



[86] I2C-bus.org. I2C – What's That? [En línea]. < http://www.i2c-bus.org/i2c-bus/>. 2016.

En cuanto a la alimentación del acelerómetro, el voltaje de alimentación será suministrado a través del pin de 3.3V cómo se puede apreciar en la Figura 38.

Es importante tener presente el resto de pines del acelerómetro, los cuales se mostraron en la Figura 23 de la sección 5.1.1.4 en dónde tenemos consecutivamente de arriba hacia abajo: GND, VCC, CS, INT1, INT2, SDO, SDA y SCL.

Teniendo en cuenta la distribución de pines, se indaga directamente en los recursos web[87] del fabricante del acelerómetro, se encuentra diagrama de conexiones y código base de ejemplo para poner en funcionamiento el producto, teniendo así la Tabla 9 en dónde se relacionan los pines que deben ser utilizados para el funcionamiento.

Tabla 9. Conexiones acelerómetro ADXL345 con Arduino UNO

Pin acelerómetro ADXL345	Pin Arduino UNO R3
GND	Tierra
VCC	3.3V
CS	3.3V
INT1	Sin uso
INT2	Sin uso
SDO	Tierra
SDA	A4
SCL	A5

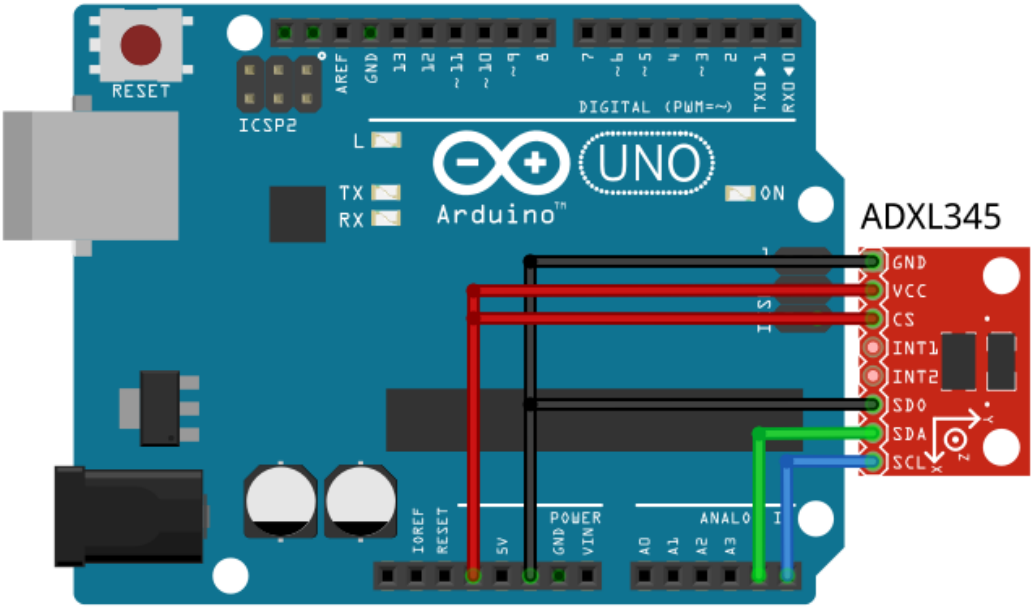
6.1.1.3 Diagrama de conexiones físicas

Partiendo de la información obtenida y las recomendaciones del fabricante del sensor, se procede a realizar las conexiones físicas del acelerómetro contra el Arduino, las cuales se pueden observar en el diagrama de conexiones mostrado en la

Figura **39**.

[87] <http://www.dfrobot.com/wiki>

Figura 39. Diagrama de conexiones Acelerómetro - Arduino UNO



6.1.1.4 Código Arduino

El código completo para operar el acelerómetro con sus respectivos comentarios se logra apreciar en el:

Anexo F. Código en Arduino del prototipo

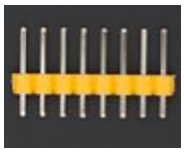
6.1.1.5 Resultados del proceso de conexiones y pruebas de funcionamiento

Se realiza la implementación práctica de la información recolectada con el fin de realizar las conexiones y pruebas iniciales del dispositivo en particular contra el Arduino.

Se lograron identificar las siguientes observaciones:

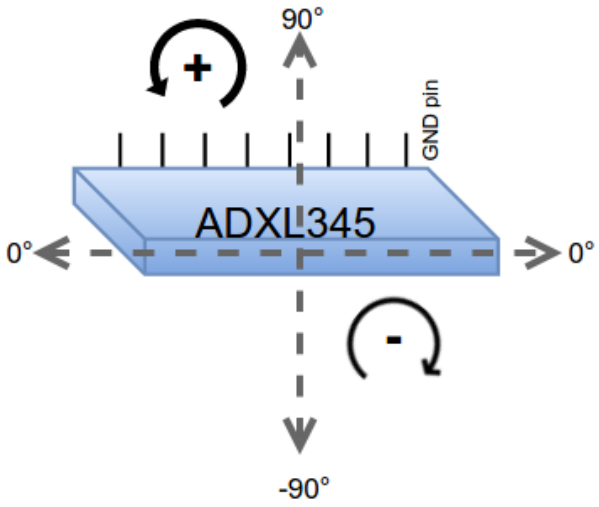
- El acelerómetro de fábrica viene con un accesorio el cual se puede ver en la Figura 40, el proceso de ensayo y error deja como conclusión que dicho accesorio debe soldarse al acelerómetro, de lo contrario no realizará contacto correctamente y no arrojará ningún valor en el monitor serie del Arduino IDE.

Figura 40. Accesorio conexión ADXL345



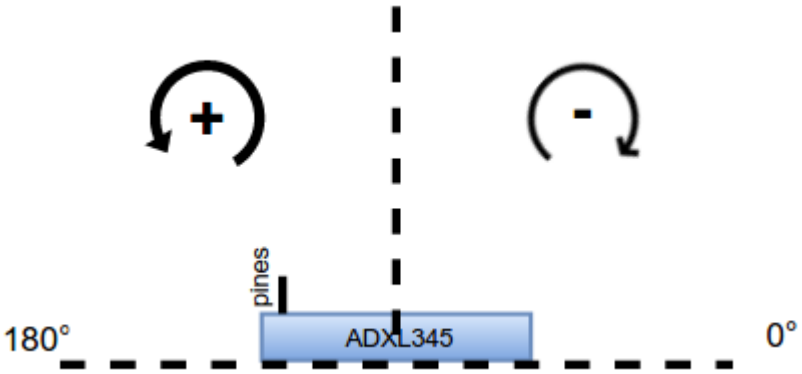
- El código ejemplo encontrado en la “wiki” del fabricante es completamente funcional, no contiene errores de programación y hace uso de la librería “Wire” del Arduino IDE que facilita la comunicación con dispositivos con bus I2C. Incluye la medida de los valores de aceleración en unidades LSB (*Least Significant Bit*) y los valores en grados de las inclinaciones que sufre el dispositivo.
- Los valores de aceleración e inclinación son bastante sensibles, se logra apreciar que el sólo hecho de tenerlo en una base sin movimiento alguno implica cambios en los valores obtenidos, los cuáles se verificarán más adelante en el modelo a escala para determinar la fiabilidad del prototipo.
- Cómo se había indicado en el párrafo 5.1.1.2 se utilizará secuencia aeroespacial: derrape (*yaw*), rodar (*roll*), cabeceo (*pitch*), el código calcula los dos últimos y sólo se utilizarán estos para la medida del movimiento. Para comprender la forma en que se comporta la medida de éstos ángulos, se ha experimentado y se logra las siguientes conclusiones iniciales:
    - El valor del pitch se comporta cómo se muestra en la Figura 41, teniendo en cuenta que el pin *GND* se encuentra a la derecha de la gráfica y dependiendo de la inclinación el valor será positivo cuando se mueva en el sentido contrario a las manecillas del reloj y negativo en el sentido de las manecillas del reloj.

Figura 41. Comportamiento del valor “Pitch”



- El valor del roll se comporta cómo se muestra en la **¡Error! La autoreferencia al marcador no es válida.**, es una vista frontal del acelerómetro, en dónde los pines se encuentran a la izquierda.

Figura 42. Comportamiento del valor “Roll”



6.1.2 Integración ultrasonido HC-SR04

6.1.2.1 Especificaciones generales a considerar del HC-SR04

Posterior a la verificación en el *datasheet* del componente, se observan las siguientes características a tener en cuenta para la integración con el Arduino:

- Voltaje de alimentación 5V
- Pin de disparador de entrada (trigger input) – digital
- Pin de eco de salida (Echo output) – digital

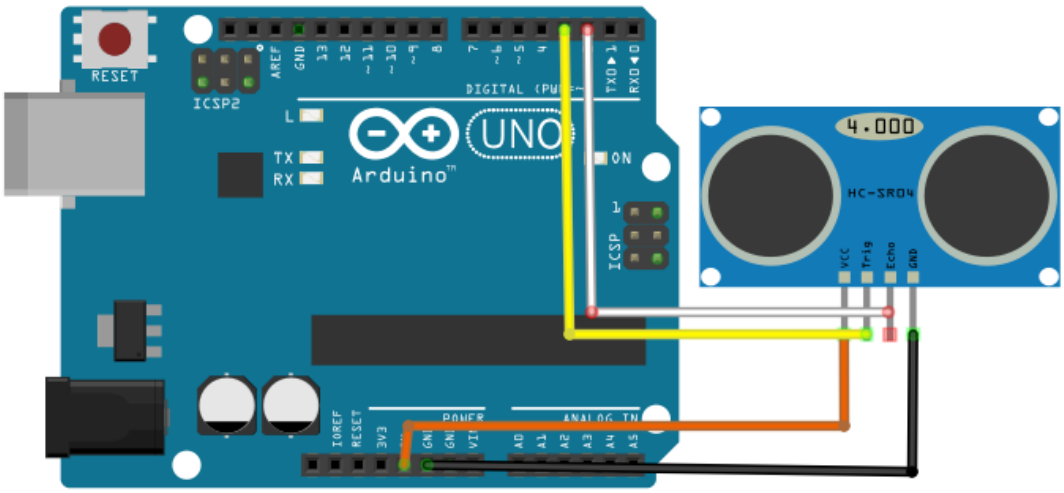
6.1.2.2 Pines a utilizar según especificaciones

La plataforma Arduino posee varias entradas para sensores de tipo digital, para este sensor en particular se utilizarán los pines 2 y 3 del grupo de entradas DIGITAL (PWM=~).

6.1.2.3 Diagrama de conexiones físicas

EL ultrasonido HC-SR04 tiene los siguientes pines para conexiones: Vcc, Trigger, Echo, GND, por ende en la Figura **43** se muestra la forma en que se conecta el dispositivo al microcontrolador Arduino.

Figura 43. Diagrama de conexiones ultrasonido – Arduino UNO.



6.1.2.4 Código Arduino

El código completo para operar el ultrasonido con sus respectivos comentarios se logra apreciar en el:

Anexo F. Código en Arduino del prototipo

6.1.2.5 Resultados del proceso de conexiones y pruebas de funcionamiento

- A través de la librería *Newping* de la biblioteca Arduino, es posible interactuar fácilmente con el sensor HC-SR04. Adicional a que es una de las librerías más precisas para trabajar con este tipo de dispositivos.

- Es necesario tener en cuenta el periodo en que se toman las medidas del sensor, dado que es necesario esperar a que el sonido sea generado, rebote y vuelva para calcular la distancia. Cómo mínimo se debe esperar 50 µs entre la generación y la recepción del sonido.
- El ángulo de toma de las medidas del ultrasonido contra la superficie de rebote debe ser menor de 15°.
- Las medidas obtenidas en primera instancia evidencian errores de precisión que es necesario suavizar con el fin de mejorar la misma. Dicho tema en particular será tratado en más detalle en la sección de pruebas de prototipo.

6.1.3 Integración módulo WiFi ESP8266

6.1.3.1 Especificaciones generales a considerar del ESP8266

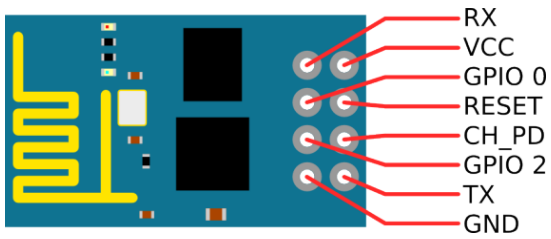
Según la hoja de datos del dispositivo, es necesario considerar los siguientes aspectos generales del módulo WiFi para conectarlo al módulo Arduino UNO.

- Voltaje de operación 3.0 V ~ 3.6 V
- Pines de Tx, Rx y CH_PD, éste último habilita o deshabilita el dispositivo dependiendo si se encuentra conectado a voltaje (HIGH) o a tierra (LOW). Dichos pines son los que se utilizan básicamente para el funcionamiento del módulo.

El mapa de pines se logra visualizar en la

- Figura **44**.

Figura 44. Pines del módulo ESP8266



6.1.3.2 Pines a utilizar según especificaciones

Es importante para este dispositivo considerar dos escenarios, el primero indica la forma en que se debe conectar el módulo para la programación del mismo a través de los comandos AT, el segundo escenario indica la forma en que se debe conectar el módulo para su operación normal posterior a la programación. Respectivamente los dos escenarios se reflejan en las Tabla 10 y en la Tabla 11.

Tabla 10. Forma de conexión del módulo ESP8266 para programación

Pin Arduino UNO R3	Pin ESP8266
RX	RX
TX	TX
GND	GND
3.3V	VCC
3.3V	CH_PD

Tabla 11. Forma de conexión del módulo ESP8266 para su funcionamiento

Pin Arduino UNO R3	Pin ESP8266
11	RX
10	TX
GND	GND
3.3V	VCC
3.3V	CH_PD

6.1.3.3 Diagrama de conexiones físicas

Debido a los dos escenarios planteados (programación y funcionamiento) se realizan los diagramas de conexiones mostrados respectivamente en la Figura **45** y

Figura **46**.

Figura 45. Diagrama de conexiones ESP8266 para programación

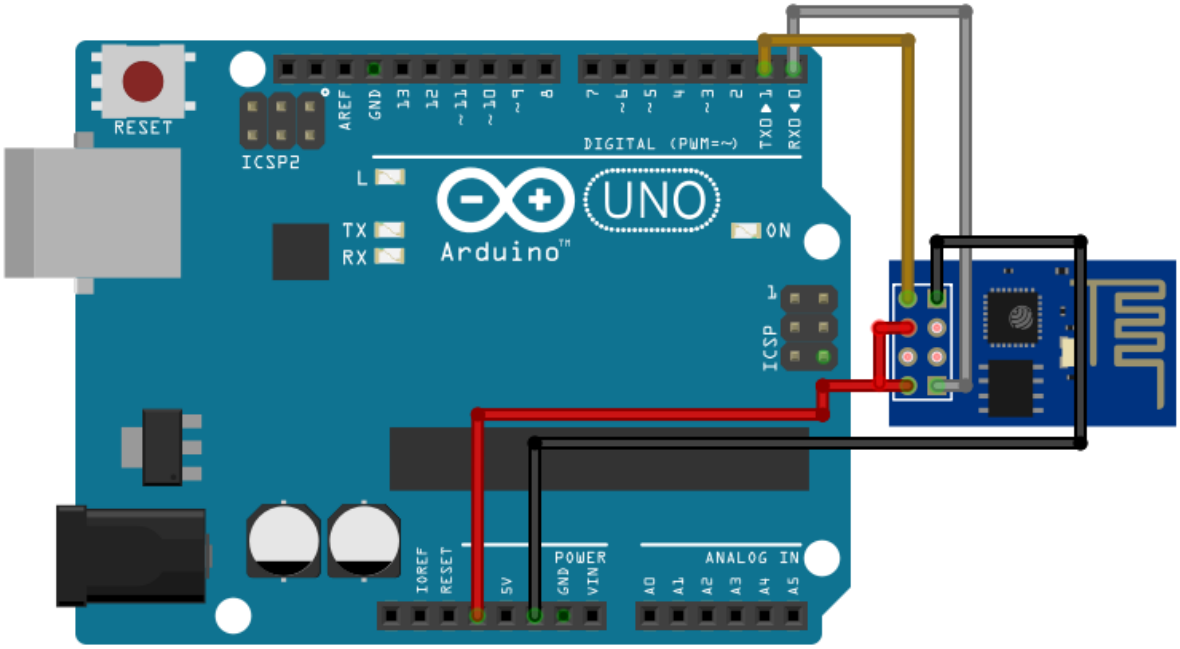
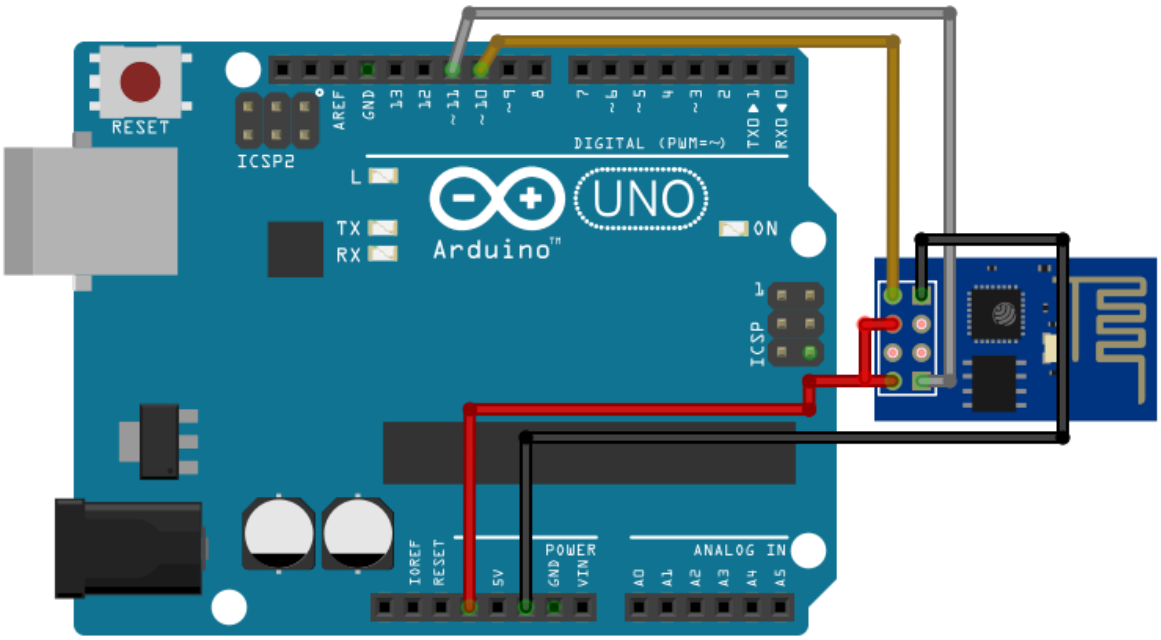


Figura 46. Diagrama de conexiones ESP8266 para funcionamiento



6.1.3.4 Código Arduino

El código completo para operar el ultrasonido con sus respectivos comentarios se logra apreciar en el:

Anexo F. Código en Arduino del prototipo

6.1.3.5 Resultados del proceso de conexiones y pruebas de funcionamiento

En el proceso de conexiones y pruebas de funcionamiento del módulo WiFi ESP8266, se tienen las siguientes observaciones y conclusiones del proceso de experimentación:

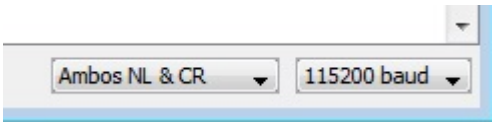
- Debido a que el acelerómetro ADXL345 opera a 3.3V y el módulo ESP8266 requiere que dos pines sean alimentados con el mismo voltaje, es necesario el uso de una protoboard dado que el Arduino sólo posee una salida de alimentación con dicho voltaje. Se obtiene una protoboard mini para no desbordar en tamaño el prototipo.
- Pruebas de la conectividad y programación
    - Posterior a la conexión de módulo, se interactúa con el módulo a través del código ejemplo del Arduino IDE denominado “*BareMinimum*” el cual contiene simplemente las siguientes líneas:

```
void setup() {
  // put your setup code here, to run once:
}
void loop() {
  // put your main code here, to run repeatedly:
}
```

En síntesis es un código en blanco para uso del módulo.

- El módulo ESP8266 viene de fábrica para establecer una comunicación serie a 115200, en primera instancia se requiere abrir el monitor en serie, en la parte inferior derecha establecer dicha velocidad de comunicación (baud) y "Ambos NL & CR" que permite el avance de nuevas líneas y el retorno automático de carro, tal y como se muestra en la Figura **47**

Figura 47. Comunicación a 115200 baud Monitor serie



- La forma más sencilla para determinar si el módulo está listo para comunicarse a través de comandos AT es enviando el comando “AT”, si el módulo responde “OK” indica que las conexiones están realizadas y el módulo está listo para recibir los comandos para su programación.
- Se verifica la guía de ejemplos de comandos AT otorgada por el fabricante Espressif para tener claridad de la forma en que deben ser ejecutados los comandos AT, se logra concluir que el módulo requiere la siguiente secuencia de comandos para el funcionamiento, comentarios del comando entre paréntesis:
  i. AT+UART_DEF=9600,8,1,0,0 (Comando para establecer el ESP8266 para comunicación a 9600 baud)
  ii. AT+CWMODE_DEF=1 (Define el modo de operación del módulo, 1:station mode, 2:softAP mode, 3:softAP + station mode)
  iii. AT+CWJAP_DEF="SSID","password" (Conexión a la red WiFi en dónde SSID es la red a conectarse y password la contraseña de la misma).

Todos los comandos aplicados se almacenan en la memoria del módulo para que no sea necesaria la ejecución de los mismos cada vez que se enciende el módulo.

La guia de comandos AT ejemplo de Espressif se puede apreciar en el:

Anexo G. AT comandos de ejemplo (esp8266 at command examples).

De igual manera, en secciones próximas se detallará la forma en que también a través de comandos AT se realiza en envío de la información.

- A diferencia del escenario de conexiones para la programación, en el escenario de funcionamiento a través del código de Arduino es necesario establecer cuáles serán lo pines de Tx / Rx y deben ser cruzados Tx con Rx y Rx contra Tx contra el módulo WiFi, es decir, según la Tabla **11** el pin 10 corresponde a Tx y el 11 a Rx.
- Cabe aclarar que la forma en que se enviará la información a través del módulo WiFi, se estará tratando en detalle en la sección 6.2.

6.1.4 Listado de elementos adquiridos

Basados en toda la información adquirida para la integración de los sensores, módulo de transmisión WiFi y plataforma de integración, a través de la Tabla **12** se evidencian los componentes adquiridos para el dispositivo.

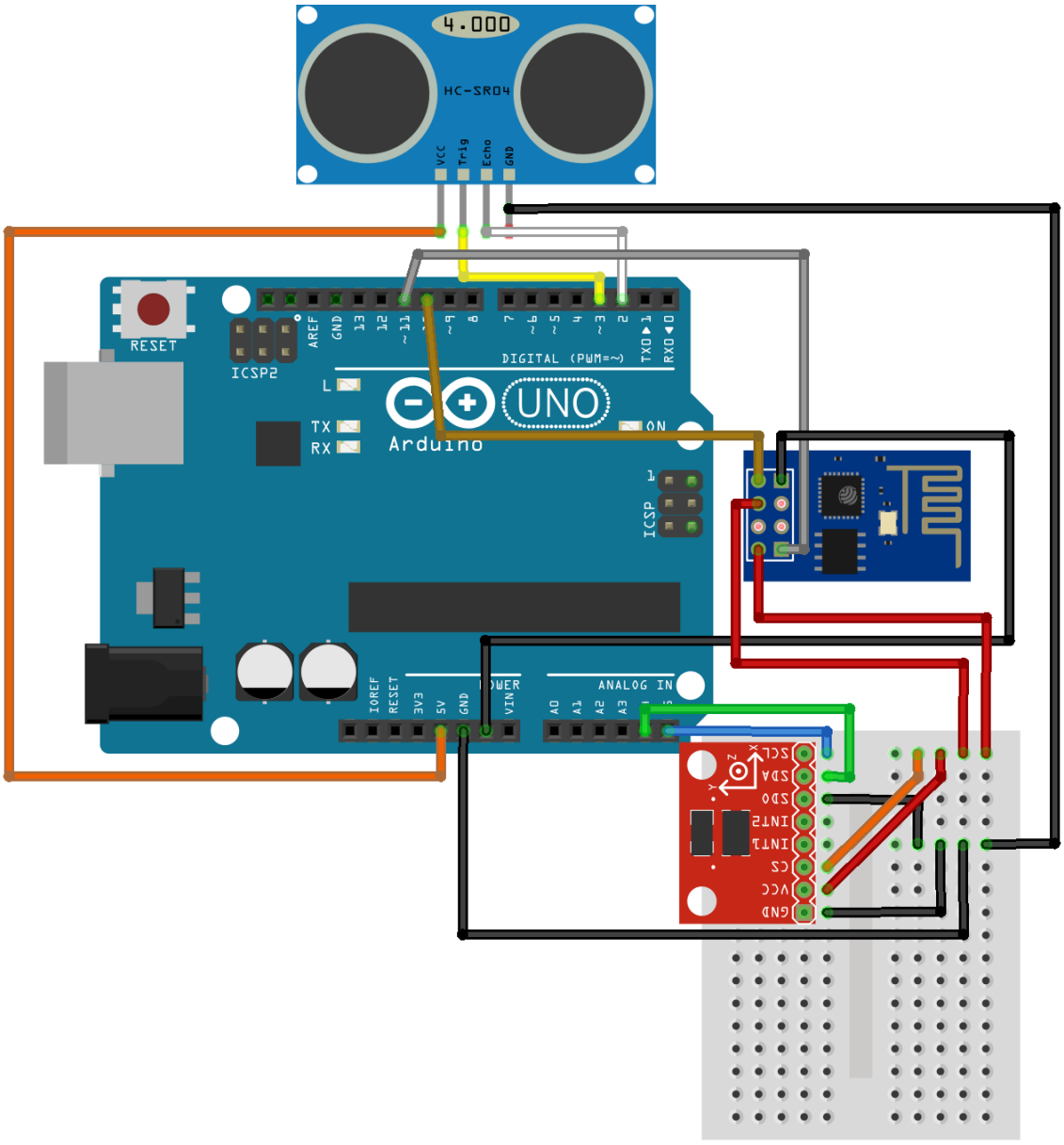
Tabla 12. Listado de elementos adquiridos

Elemento	Descripción
1 Arduino UNO R3	Plataforma de integración
1 Módulo ESP8266	Módulo de transmisión WiFi
1 Acelerómetro ADXL345	Sensor medición Mov. Horizontales
1 Ultrasonido HC-SR04	Sensor medición Mov. Verticales
1 mini protoboard	Protoboard conexión de dispositivos
1 caja plástica par Arduino	Caja para insertar el Arduino

6.1.5 Diagrama completo de conexiones y prototipo conectado

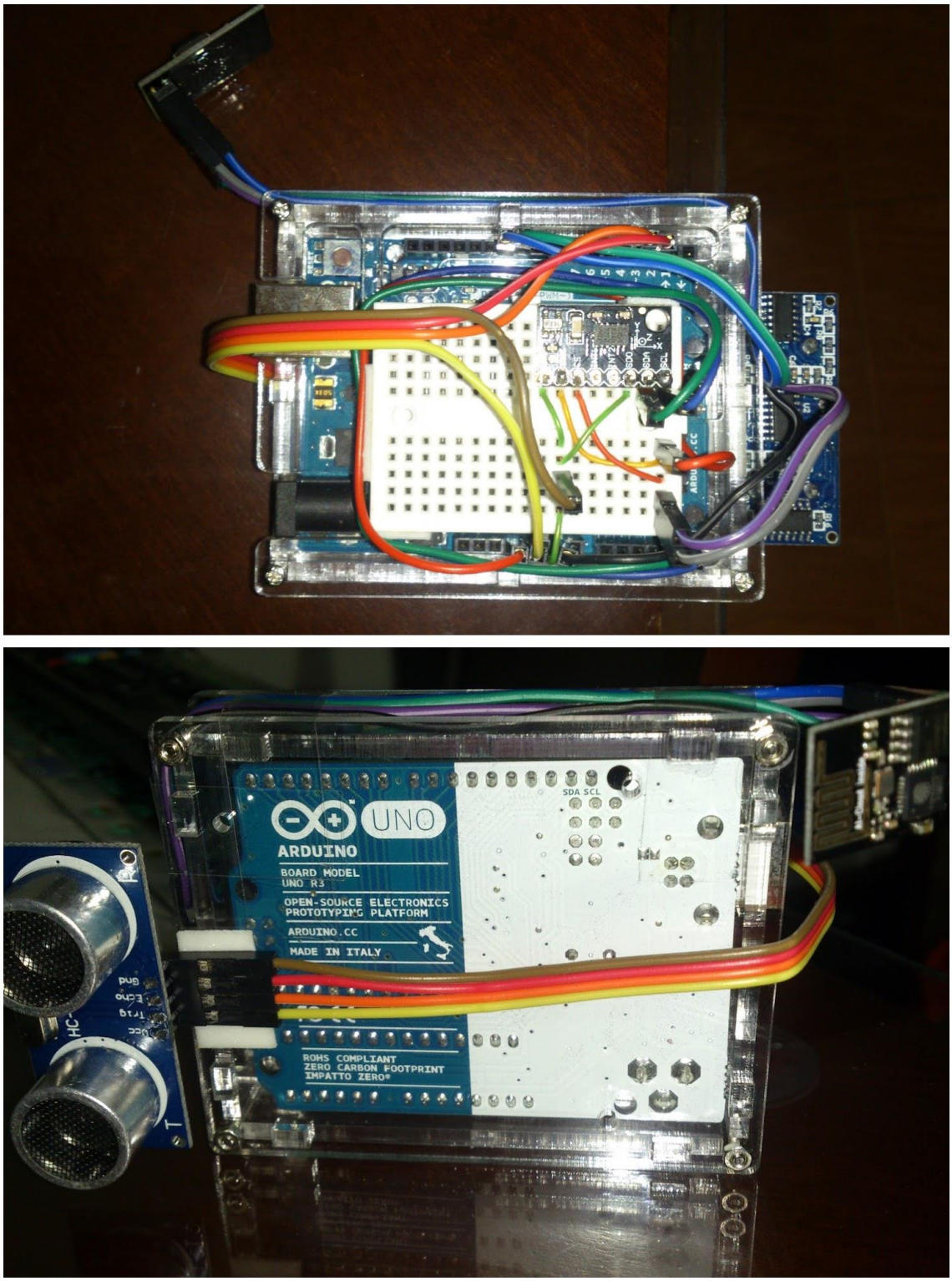
Según la forma en que se deben conectar los dispositivos, se tiene el siguiente diagrama de conexiones completo:

Figura 48. Diagrama de conexiones completo



Las conexiones fueron realizadas en el Arduino Uno, la forma en que quedaron las conexiones y el prototipo se puede apreciar en la Figura 49.

Figura 49. Prototipo conectado



6.2 ALMACENAMIENTO Y TRATAMIENTO DE LA INFORMACIÓN RECOLECTADA

Retomemos el significado de telemetría: “Conjunto de técnicas para la medida a distancia de magnitudes físicas”[88].
Ya se tienen las medidas físicas y se eligió la tecnología en que será transportada la información, es necesario enfocarse en la forma en que dicha información obtenida, será mostrada, y si requiere previa manipulación o no.
En esta parte se muestra la forma en que se llegará a cumplir éste objetivo en particular.

6.2.1 Internet de las cosas (IoT)

Se ha optado por tener presente el concepto de internet de las cosas, que implica la interacción de objetos, sensores y demás con internet, pero con algo adicional e importante a la vez, sin interacción humana que determine la medición y si la información será enviada o no. Citamos a Kevin Ashton quien claramente lo manifestó en el 2009:

“Aunque, la tecnología de la información actual es tan dependiente de los datos escritos por personas que nuestros ordenadores saben más sobre ideas que sobre cosas. Si tuviéramos ordenadores que supieran todo lo que tuvieran que saber sobre las “cosas”, mediante el uso de datos que ellos mismos pudieran recoger sin nuestra ayuda, nosotros podríamos monitorizar, contar y localizar todo a nuestro alrededor, de esta manera se reducirían increíblemente gastos, pérdidas y costes. Sabríamos cuando reemplazar, reparar o recuperar lo que fuera, así como conocer si su funcionamiento estuviera siendo correcto. El internet de las cosas tiene el potencial para cambiar el mundo tal y como hizo la revolución digital hace unas décadas. Tal vez incluso hasta más.” [89]

Una empresa significativa de las telecomunicaciones como Cisco systems, estima que para 2020 habrá 50 billones de dispositivos conectados a internet[90]. De igual forma, dicha empresa ve la importancia del concepto y ya dispone de productos, soluciones y servicios orientados a IoT.

Con el fin de indicar el crecimiento del concepto de IoT, se ha realizado la búsqueda en Google trends del término Internet de las cosas, teniendo los siguientes resultados a nivel mundial:

---

[88] RAE, Telemetría, [En línea]. <http://dle.rae.es/?id=ZN0Lqtu>. 2016.

[89] Wikipedia, Internet de las cosas, [En línea]. <https://es.wikipedia.org/wiki/Internet_de_las_cosas>.2016.

[90] Cisco systems, Internet of things, [En línea]. <http://www.cisco.com/c/en/us/solutions/internet-of-things/overview.html?>.2016

Figura 50. Interés a lo largo del tiempo del campo de estudio – Internet de las cosas.[91]

Figura 51. Interés por región[92]

Se verifica ahora la tendencia por el país, para tener una medida de la tendencia en Colombia:

Figura 52. Interés a lo largo del tiempo del campo de estudio – Internet de las cosas en Colombia[93]

[91] Google trends, [En línea], <https://www.google.com/trends/.>. 2016.

[92] Google trends, [En línea], <https://www.google.com/trends/.>. 2016.

[93] Google trends, [En línea], <https://www.google.com/trends/.>. 2016.

Figura 53. Interés por región en Colombia[94]

Los diagramas y los mapas de la tendencia indican el alza que tiene el concepto tanto para el país, como para el mundo. Lo que se pretende para el dispositivo prototipo es que tenga las bondades que el Internet de las cosas ofrece.

6.2.2 ThingSpeak

En la búsqueda de tener una plataforma orientada al Internet de las cosas, se ha encontrado y elegido a Thingspeak para la recepción y el trato de la información de las medidas obtenidas por el prototipo, se le da un especial trato dado que se describirán las posibilidades de la plataforma y se explicará el proceso que llevó para orquestarlo a lo ya realizado con la integración de los sensores.

6.2.2.1 Descripción, características y ventajas observadas para la elección.

ThingSpeak es una plataforma de Internet de las cosas que permite recoger datos de los sensores y almacenar información en la nube. La plataforma ThingSpeak ofrece aplicaciones que permiten analizar y visualizar los datos en MATLAB, y luego actuar sobre los mismos. Los datos de los sensores se pueden enviar a ThingSpeak a través de Arduino, Raspberry Pi, BeagleBone Black, y otro hardware.[95]

Características generales de ThingSpeak:

- Recolección y almacenamiento de información en tiempo real
- Analítica y visualizaciones con MATLAB
- Alertas y programaciones
- Comunicación de dispositivos
- API de código abierto
- Geo localización de información

Características adicionales – Apps.

---

[94] Google trends, [En línea], <https://www.google.com/trends/.>. 2016.

[95] MathWorks, Thingspeak, [En línea], < https://www.mathworks.com/help/thingspeak/>. 2016.

Los canales en ThingSpeak almacenan datos. Los datos pueden ser enviados desde la Web o a través de dispositivos a un canal en ThingSpeak. A través de las aplicaciones de la Tabla 13 ThingSpeak está en la capacidad de transformar/mostrar los datos o disparar una acción.

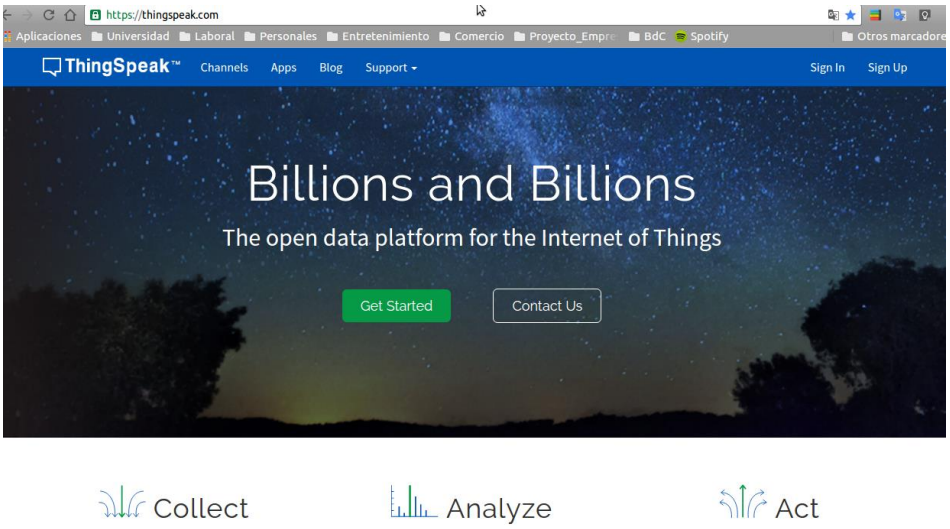
Tabla 13. Aplicaciones en ThingSpeak

Tipo	Característica	Descripción / Capacidades
Análisis	MATLAB Analysis	Explora y transforma la información almacenada en un canal.
	MATLAB Visualizations	Permite visualizar la información en gráficas de MATLAB.
	Plugins	Permite el uso de medidores de Google o de complementos escritos por desarrolladores.
Acciones	ThingTweet	Conecta un dispositivo a Twitter y envía alertas
	TweetControl	Permite enviar mensajes por twitter con ciertas palabras para que actúen cómo disparadores en tiempo real.
	TimeControl	Permite a las aplicaciones de ThingSpeak ser programadas para ser ejecutadas.
	React	Reacciona cuando un canal encuentra ciertas condiciones.
	TalkBack	Encola comandos a un dispositivo remoto para controlarlo.
	ThingHTTP	Permite la comunicación entre dispositivos, sitios web y servicios web sin tener que implementar el protocolo a nivel de módulo

Tal y cómo lo menciona su sitio web, Thingspeak es una plataforma de código abierto para el internet de las cosas (

Figura **54**).

Figura 54. Thingspeak.com



Posee la bondad de las plataformas de código abierto, compatibilidad multiplatafoma, funciona en línea con uso de la infraestructura del sitio, también con la posibilidad de descargar los binarios e implementarla en equipos localmente, foro oficial en la página, documentación oficial y abierta, tutoriales, ejemplos, aplicaciones para interactuar con la información y sin costo alguno de uso.Por estas razones se optó por experimentar con dicha plataforma para que reciba la información y explorar las posibilidades de ThingSpeak.

6.2.2.2 Interacción inicial con la plataforma

A través de la guía de pasos iniciales del sitio web oficial de ThingSpeak, se experimenta la forma en que se debe cargar la información al sitio, teniendo los siguientes pasos a seguir:

i. Crear una cuenta en ThingSpeak.com e iniciar sesión con la misma.
ii. En el menú principal “Channels” se debe crear un canal que se encargará de recibir la información.
iii. En formulario de diligenciamiento para el nuevo canal, se le debe asignar un nombre al mismo, se debe también seleccionar la cantidad de campos/variables en los que se almacenará la información y adicional asignar una descripción de cada uno.
iv. Guardar el canal dando clic en el botón “Save channel”.
v. Al crear el canal, es necesario tener presente dos datos relevantes, uno es el ID del canal (Channel ID) y el otro son las llaves API (API Keys), de las cuales existen una de lectura y una de escritura por canal. A través de estos datos se realiza la interacción con los canales, al subir o extraer información de los mismos.

A través de los pasos anteriores, se establece lo mínimo y necesario para empezar a subir información a ThingSpeak, es importante tener claridad acerca del ID del canal y de las llaves de lectura y escritura, dado que dicha información es la que deberemos invocar en el código del Arduino para poder subir la información capturada por los sensores.
La guía detallada de los pasos iniciales en ThingSpeak se lograr observar en el:
Anexo H. Pasos iniciales en ThingSpeak

6.2.3 Implementación de ThingSpeak en el prototipo.

6.2.3.1 Pruebas de envío de información.

Posterior a la interacción inicial, se constituye lo necesario para poder subir información al canal de la plataforma, de igual forma se busca en la Web lo necesario para realizarlo a través de Arduino y el módulo de transmisión ESP8266 dado nuestro caso puntual.
Las pruebas iniciales se realizan basados en el Tutorial: "*IoT Datalogger with ESP8266 WiFi Module and FRDM-KL25Z*[96]" encontrado en internet obteniendo para nuestro caso en particular los siguientes resultados:

- Se trabaja con el canal de prueba creado en el apartado anterior (6.2.2.2) el cual tiene el ID 145110, las llaves de escritura/lectura "EZX4MABY3305EUYE" y "GSFO0FU51YUG4QN4".
- Se asume que el módulo ESP8266 ya tiene almacenado en la memoria el SSID y contraseña para conectarse a la red WiFi ya tratados en la sección 6.1.3.5.
- Es necesario establecer las conexiones del módulo WiFi según el escenario de programación ya mostrado en la Figura 45 y probarla enviando el comando *AT* a través del software para el Arduino, el módulo debe responder *OK* lo que indica que está apto para recibir el resto de comandos.
- La secuencia de comandos para probar el envío de la información de parte del módulo ESP8266 y la recepción de parte del canal en ThingSpeak que se muestra en la Figura 55 se explican en la Tabla 14, cabe aclarar que en la tabla se muestra un comando adicional posterior al comando 3, y es la información enviada al canal.

Tabla 14. Comandos de prueba para envío de información ThingSpeak - ESP8266

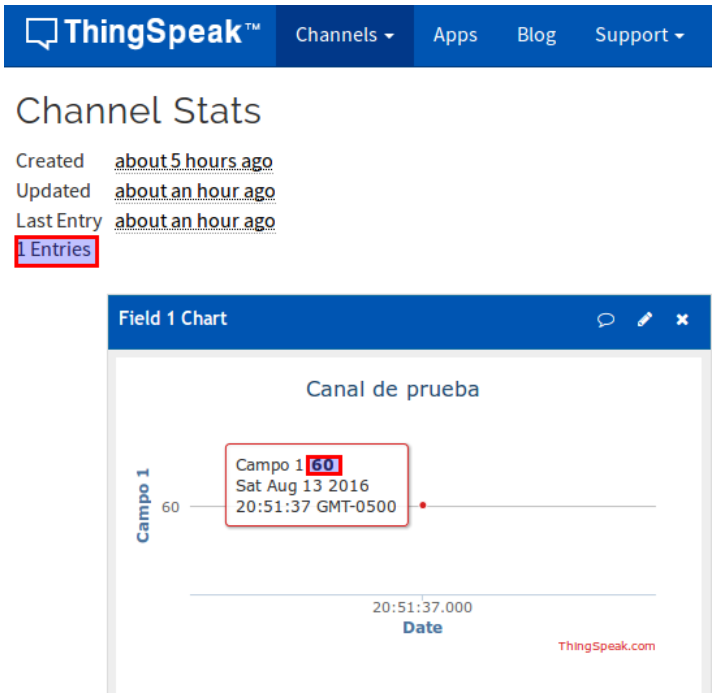
#	Comando	Descripción - Uso
1	AT+CIPMUX=1	Habilita las múltiples conexiones al módulo WiFi
2	AT+CIPSTART=4,"TCP","184.106.153.149",80	Inicia una conexión TCP con la dirección IP pública de Thingspeak a través del puerto 80
3	AT+CIPSEND=4,44	Se le indica a la conexión la cantidad de caracteres que serán enviados en la cadena, 44 para la prueba. Al recibir el comando > indica que la conexión está lista para recibir la información.
4	GET /update?key= EZX4MABY3305EUYE &field1=60	Se envía la cadena de caracteres, indicando la llave de escritura para el canal, el campo específico (field1) y el valor a cargarle al campo (60).

Figura 55. Secuencia prueba ESP8266 – ThingSpeak

[96] Erich Styger, IoT Datalogger with ESP8266 WiFi Module and FRDM-KL25Z, [En línea], < https://mcuoneclipse.com/2014/12/14/tutorial-iot-datalogger-with-esp8266-wifi-module-and-frdm-kl25z/>. 2014.

- Posterior a recibir la confirmación de la información enviada, se verifica el canal de prueba generado y se comprueba la correcta recepción del dato enviado (60) al campo 1 tal como se muestra en la Figura **56**. La generación de la gráfica es automática, simplemente registra el dato recibido con la marca del tiempo.

Figura 56. Carga de información en el canal de prueba



6.2.3.2 Envío de información de los sensores.

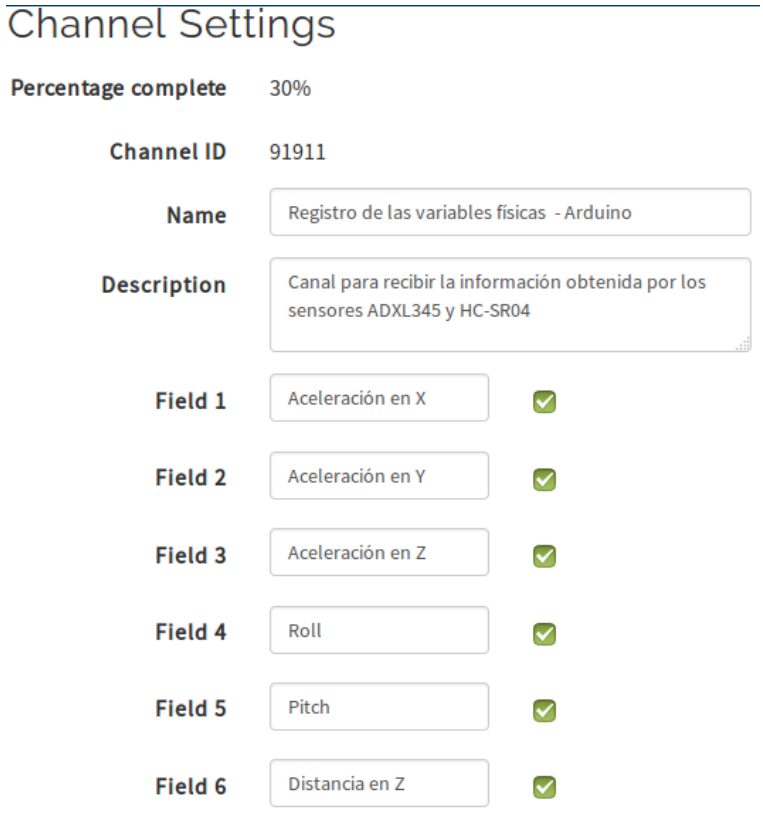
Posterior a realizar las pruebas de carga de información se orienta el proceso de experimentación a modelar de forma muy puntal el código y los comandos con los cuales se debía enviar la información recolectada en tiempo real por los sensores a ThingSpeak.

En este apartado se detallará la forma en que se definieron los canales para la recepción, también la forma en que se verificó el código y se ajustó para los datos obtenidos.
Igualmente se muestran los resultados obtenidos con la carga de información en la plataforma web de ThingSpeak.

6.2.3.2.1 Canal de recepción inicial de las variables.

En primera instancia, se define un canal para la recepción de los datos obtenidos por los sensores, se crea el canal con nombre “Registro de las variables físicas - Arduino”, se habilitan 6 campos para la recepción de los datos, los detalles de estos se visualizan en la Figura **57**.

Figura 57. Canal en ThingSpeak para la recepción de los datos



El canal tiene como llave de escritura: BYIO25TYJ1OVGMUP y como llave de lectura: 3PD43392VBXM4WE0.

6.2.3.2.2 Consideraciones del código para la carga de información.

Teniendo en cuenta los comandos que se requieren para subir la información mostrados en la Tabla 14, se trata de forma particular la forma en la cual deben ser armonizados para que envíen la información de cada uno de los sensores, para ello es importante tener en cuenta lo siguiente acerca de las medidas físicas obtenidas:

- La forma de conectar el módulo ESP8266 debe ser la planteada en el escenario 2 mostrado en la Figura 46.
- El código para el funcionamiento del acelerómetro ADXL345 tiene 5 variables en formato numérico para el almacenamiento de las aceleraciones y las inclinaciones, X, Y, Z corresponden a los valores de aceleración en cada uno de los ejes, *roll* y *pitch* a los valores de inclinación.
- El código para el funcionamiento del ultrasonido HC-SR04 tiene una variable en formato numérico para el almacenamiento de la distancia.

Con estas consideraciones previas, el código en específico con el que se envía la información del primer canal en resumen realiza:

i. Declarar de forma global una variable de tipo String para almacenar la llave de escritura (*Write API Key*).
ii. Convertir los valores numéricos enteros en cadenas de caracteres a través de la función dtostrf.
iii. Establecer la conexión hacia la plataforma concatenando el comando AT+CIPSTART con la dirección IP pública de ThingSpeak y el puerto http (80).
iv. Contabilizar la longitud y preparar la cadena de caracteres para enviar la información a través del comando GET concatenando los valores del paso ii.
v. Condicional, Si posterior al enviar el comando AT+CIPSTART el módulo responde con el valor '>' enviar la información a cada uno de los 6 campos de forma consecutiva.

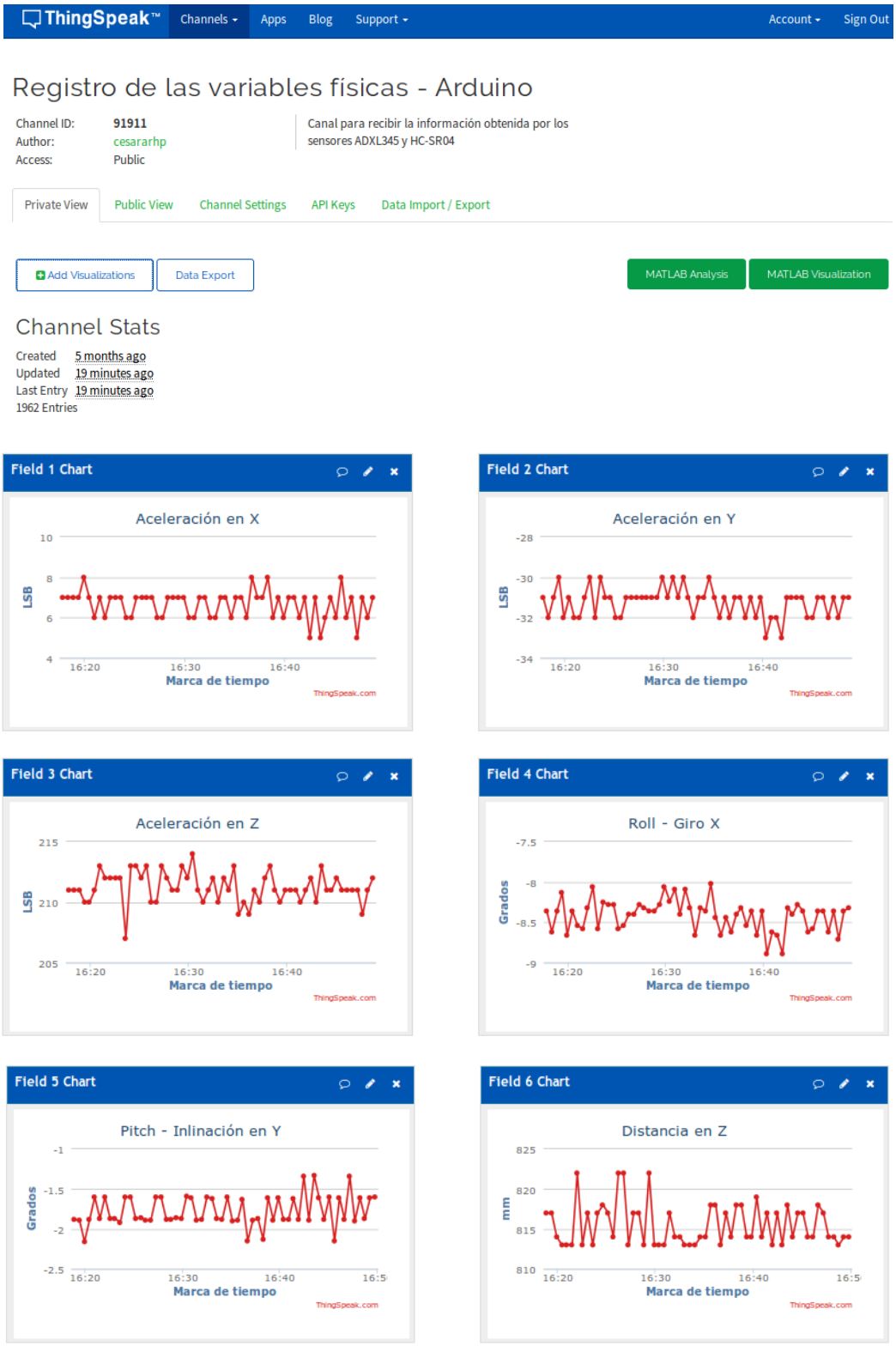
El código completo para la carga de información a través de ThingSpeak con sus respectivos comentarios se logra apreciar en el **Anexo F. Código en Arduino del prototipo**

6.2.3.2.3 Registro de las medidas obtenidas y evidencias.

Posterior a cargar el código en el Arduino, se comprueba que el canal está recibiendo información de forma satisfactoria. Las evidencias de la recepción correcta se logran apreciar respectivamente en la

Figura **58**.

Figura 58. Información de los sensores en el canal de ThingSpeak.



6.2.3.3 Tratamiento de la información almacenada.

Ya se conocen las posibilidades que tiene ThingSpeak para el tratamiento de la información, teniendo en cuenta estas capacidades de la plataforma, se opta por usarlas y explorar los resultados. Estas manipulaciones en particular serán tratadas individualmente, para explicar lo que se pretende, detallar lo necesario para llegar a ello y mostrar los resultados.

6.2.3.3.1 Registro de las diferencias entre medidas.

Posterior al registro de las variables físicas en el canal en ThingSpeak y según como se pudo apreciar en la

Figura **58**, el canal registra cada uno de los valores en tiempo real del prototipo.
Evaluando la forma en que se visualiza la información, se determina la necesidad de tener claridad acerca de la variación entre medidas, es decir, al momento de presentarse un primer registro e inmediatamente al presentarse un segundo, calcular y registrar los cambios entre las medidas.
Para ello, es necesario utilizar las *Apps* de la Tabla **13**, comprender las posibilidades e indagar la forma en que se debe llegar a dicho resultado. Se sigue la documentación oficial alojada en la página de Thingspeak para la aplicación denominada *MATLAB Analysis*, y en especial el tutorial para análisis de información[97].
Teniendo dicha claridad, para generar el registro entre las diferencias, en resumen fue necesario:

i. Comprender que *MATLAB Analysis* requiere que la información tratada ya se encuentre previamente registrada en un canal de ThingSpeak.
ii. El resultado de dicho análisis, debe ser almacenado en un canal diferente al del origen de la información, por ende, se crea el canal “Registro de las diferencias”, al cual se le habilitan los campos mostrados en la Figura 59.
iii. Modelar el código en MATLAB para realizar el análisis de la información y almacenar los resultados en el canal del paso ii.
iv. Se indagó una función en MATLAB para el cálculo de la diferencia entre medidas, encontrándose la función diff, cuya descripción de funcionamiento se visualiza en la Tabla **15**.
v. Posterior a tener el resultado esperado, se valida que el código en ThingSpeak debe ser programado para ejecución periódica, de lo contrario sería necesario la ejecución manual, para solventarlo, se usa la App *TimeControl* de la plataforma.

El detalle paso a paso para lograr el registro correcto de las diferencias y el uso de *TimeControl* es posible verlo en el:
Anexo I. *MATLAB Analysis* y *TimeControl* para calcular las diferencias entre medidas.

Figura 59. Canal para el registro de las diferencias

[97] Mathworks, Help, [En línea], <https://www.mathworks.com/help/thingspeak/analyze-your-data.html>. 2016.

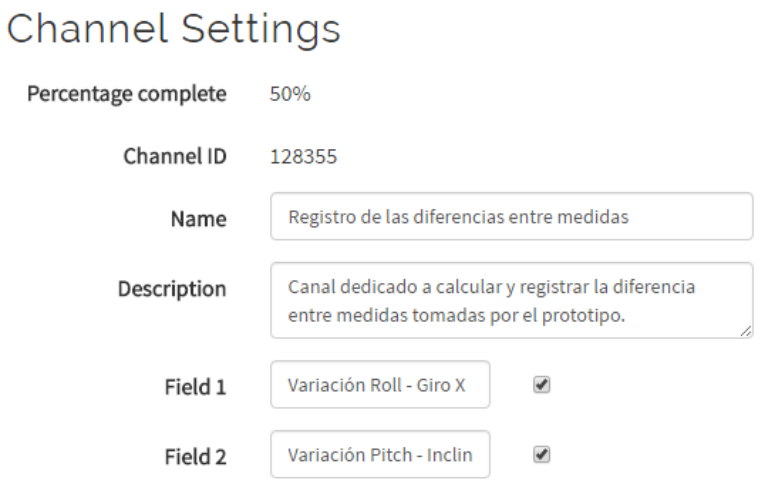


Tabla 15. Detalles de la función diff en MATLAB[98].

Función	Sintaxis	Descripción	Ejemplo
	Y = diff(X)	Y = diff(X) calcula las diferencias entre elementos adyacentes de X a lo largo de la primera dimensión de matriz cuyo tamaño no es igual a 1. Si X es un vector de longitud m, Y = diff(X) devuelve un vector de longitud m-1. Los elementos de Y son las diferencias entre elementos adyacentes de X. Y = [X(2)-X(1) X(3)-X(2) ... X(m)-X(m-1)]	X = [1 1 2 3 5 8 13 21]; Y = diff(X) Y = 0 1 1 2 3 5 8

El código en MATLAB para realizar el análisis de la información, el cálculo de las diferencias y el registro en el segundo canal con sus respectivos comentarios (en naranja), se muestra a continuación:

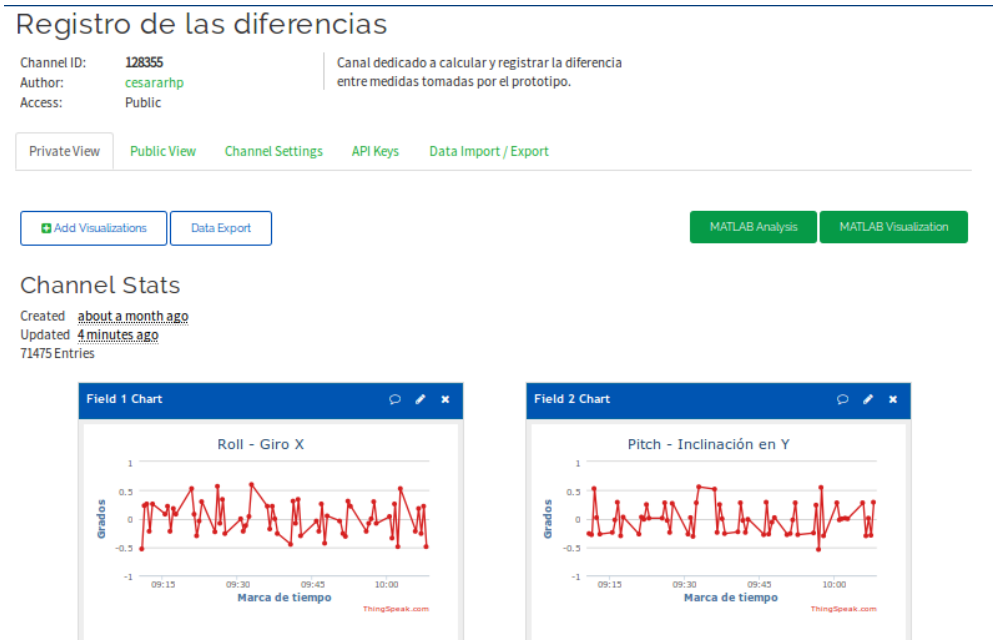
```
readChannelID = 91911; %Declaración de constante con el ID del canal origen de la información
readAPIKey = '3PD43392VBXM4WE0'; %Declaración de constante con la llave de lectura del canal origen de la información
writeChannelID = 128355; %Declaración de constante con el ID del canal destino donde se registrara el resultado del análisis
writeAPIKey = '8W72PYK32YLRMZGI'; %Declaración de constante con la llave de escritura del canal destino
```

[98] Mathworks, diff, [En línea], <http://www.mathworks.com/help/matlab/ref/diff.html>. 2016.

```
writeChannelID2 = 145110; %Declaración de constante con el ID del canal destino donde se registrara el resultado del análisis
writeAPIKey2 = 'EZX4MABY3305EUYE'; %Declaración de constante con la llave de escritura del canal destino
[dist,time] =
thingSpeakRead(readChannelID,'Fields',6,'NumPoints',6,'OutputFormat','matrix');
%Declaración de dos variables [dist,time] para leer  y almacenar del canal origen los últimos 6 valores del campo 6 con su respectiva marca de tiempo y que su formato de salida sea una matriz
dist2= thingSpeakRead(readChannelID,'Fields',7,'NumPoints',6);%Declaracion de una variable [dist2] para leer  y almacenar del canal origen los últimos 6 valores del campo 7
roll= thingSpeakRead(readChannelID,'Fields',4,'NumPoints',6); %Declaración de una variable [roll] para leer  y almacenar del canal origen los últimos 6 valores del campo 4
pitch= thingSpeakRead(readChannelID,'Fields',5,'NumPoints',6); %Declaración de una variable [pitch] para leer y almacenar del canal origen los últimos 6 valores del campo 5
varD=(diff(dist)); %Declaración de variable para almacenar la diferencia de los valores almacenados en dist
varD2=diff(dist2);%Declaración de variable para almacenar la diferencia de los valores almacenados en dist2
varR=diff(roll); %Declaración de variable para almacenar la diferencia de los valores almacenados en roll
varP=diff(pitch); %Declaración de variable para almacenar la diferencia de los valores almacenados en pitch
dataTable = table(time(2:6,1:1),varR,varP); %Generación de una tabla con los cálculos de las diferencias entre valores del roll y pitch
dataTable2 = table(time(2:6,1:1),varD,varD2); %Generación de una tabla con los cálculos de las diferencias entre valores de la distancia en Z
disp(dataTable); %Comando para mostrar la tabla generada
disp(dataTable2); %Comando para mostrar la tabla generada
thingSpeakWrite(writeChannelID, dataTable,'WriteKey',writeAPIKey) %Comando para escribir los datos de la tabla en el canal destino, en campos sucesivos del mismo.
thingSpeakWrite(writeChannelID2, dataTable2,'WriteKey',writeAPIKey2) %Comando para escribir los datos de la tabla en el canal destino, en campos sucesivos del mismo.
```

El funcionamiento y registros del canal es posible verlo en la Figura **60**, allí se logra observar que el cambio entre medidas es bajo debido a que el prototipo no está sufriendo movimiento alguno que indique un cambio drástico entre medidas.

Figura 60. Canal "Registro de las diferencias" en ThingSpeak.



En las gráficas anteriores, se logra ver que la variación de valores tales como el Roll y el Pitch está en el rango de -0.5 y 0.5.
La distancia en Z medida está presentando variaciones aleatorias entre medidas incluso con el prototipo completamente estático, cómo ya se había mencionado previamente, dicho tema se tratará en las pruebas del prototipo en el modelo a escala. Ésta última información (variación de la distancia en Z) se almacenará en un tercer canal para tener en uno sólo la consolidación de todo el movimiento.

6.2.3.3.2 Cálculo de la distancia basado en el ángulo de inclinación.

Cómo fue indicado en el apartado 4.2.1, la medición de la deriva será realizada a través de funciones trigonométricas y la cantidad de movimiento registrada entre dos medidas del prototipo, ésta última tratada en el apartado anterior (6.2.3.3.1). Se usará Thingspeak para realizar dicho cálculo y así registrar la cantidad total de movimiento; utilizaremos de nuevo el App *MATLAB Analysis* para tratar la información y graficarla.
En síntesis, para calcular dicha distancia, fue necesario:

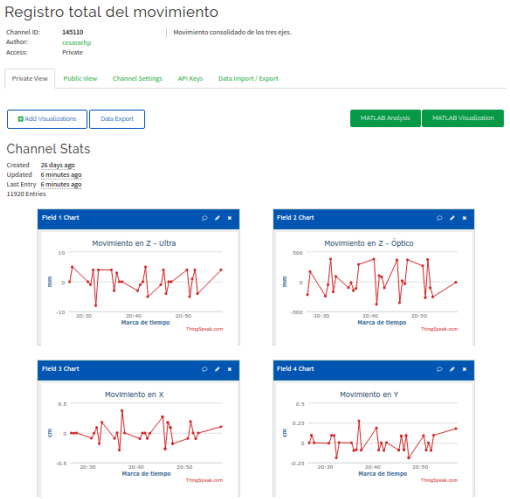
i. Modelar el código para tomar la variación entre medidas de los ángulos pitch & roll y calcular el desplazamiento.
ii. Crear un tercer canal en ThingSpeak para almacenar el resultado total del movimiento con nombre
iii. Crear una tarea recurrente mediante TimeControl para la ejecución automática del código modelado.

El código en MATLAB para realizar el análisis de la información, el cálculo de los desplazamientos y el registro en el tercer canal con sus respectivos comentarios (en naranja), se muestra a continuación:

```
readChannelID = 128355; %Declaración de constante con el ID del canal origen de la información
readAPIKey = '92NTCACXGFZUNJHI'; %Declaración de constante con la llave de lectura del canal origen de la información
writeChannelID = 145110; %Declaración de constante con el ID del canal destino donde se registrara el resultado del análisis
writeAPIKey = 'EZX4MABY3305EUYE'; %Declaración de constante con la llave de escritura del canal destino
[X,time] = thingSpeakRead(readChannelID,'Fields',1,'NumPoints',5); %Declaración de dos variables [X,time] para leer  y almacenar del canal origen los últimos valores del campo 1 con su respectiva marca de tiempo
[Y] = thingSpeakRead(readChannelID,'Fields',2,'NumPoints',5); %Declaración de una variable [dist2] para leer  y almacenar del canal origen los últimos valores del campo 2
distX = tand(X)*(24); %Declaración de la variable distX para almacenar el resultado del cálculo de la distancia por la función tangente y el dato previamente adquirido
distY = tand(Y)*(24); %Declaración de la variable distY para almacenar el resultado del cálculo de la distancia por la funcion tangente y el dato previamente adquirido
disp(distX); %Muestra de los resultados de los dos cálculos
disp(distY); %Muestra de los resultados de los dos cálculos
thingSpeakWrite(writeChannelID,[distX,distY],'Fields',[3,4],'WriteKey',writeAPIKey,'TimeStamp',time) %Escritura en el tercer canal de ThingSpeak
```

El funcionamiento y registros del canal es posible verlo en la Figura **61**.

Figura 61. Canal " Registro total del movimiento" en ThingSpeak



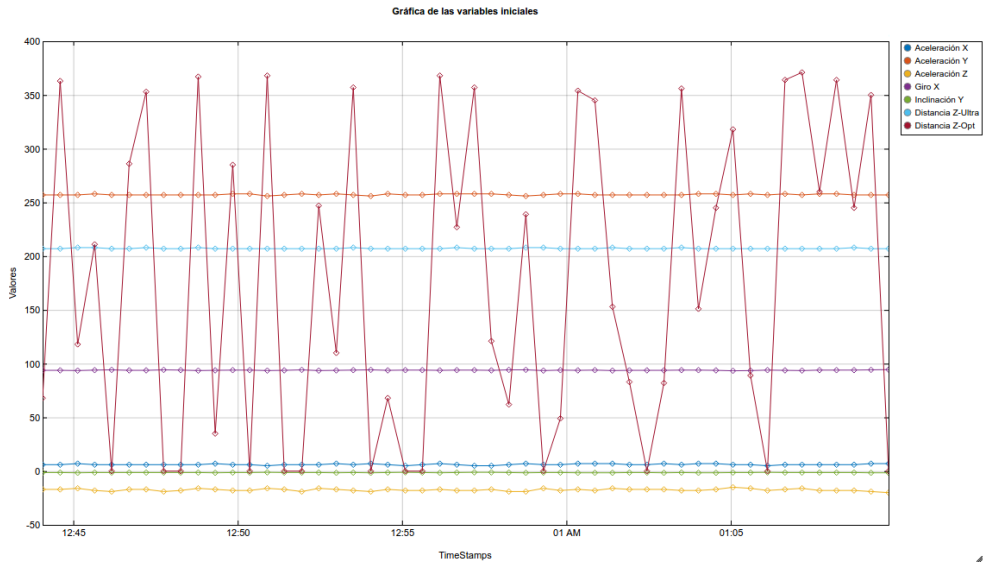
6.2.3.3.3 Thingplot – gráficas adicionales.

Complementando cada uno de los canales de ThingSpeak creados para la visualización y análisis de la información, ahora se explora otra de las aplicaciones embebidas de la plataforma: MATLAB Visualizations.
Con dicha aplicación, es posible unificar todas las gráficas de un canal en una sola con el fin de analizar la información de forma más sencilla, simplemente debemos modelar el código con el cuál leeremos la información almacenada en los canales y graficamos dicha información.
El código en MATLAB para realizarlo con sus respectivos comentarios (en naranja), se muestra a continuación:

```
readChId = 91911 %ID del canal origen de la información
readKey = '3PD43392VBXM4WE0' %Llave de lectura del canal origen de la información
[Datos,tiempo] = thingSpeakRead(readChId,'fields',[1,2,3,4,5,6,7],... %lectura de los campos del canal
    'NumPoints',50,'ReadKey',readKey); %Cantidad de registros (50) a leer de los campos
thingSpeakPlot(tiempo,Datos,'xlabel','TimeStamps',... %Comando para graficar los datos leídos
    'ylabel','Valores','title','Gráfica de las variables iniciales',... %Asignación de título a la gráfica
    'Legend',{'Aceleración X','Aceleración Y','Aceleración Z','Giro X','Inclinación Y','Distancia Z-Ultra','Distancia Z-Opt'},'grid','on','Marker','d','LineStyle','-','LineWidth',1); %Leyenda para cada uno de los campos y otras opciones de la línea
```

La muestra de una de las gráficas generadas es posible apreciarla en la Figura **62**.

Figura 62. Gráfica realizada con MATLAB Visualizations.



El detalle paso a paso para lograr el registro correcto de las diferencias y el uso de *TimeControl* es posible verlo en el:
Anexo J. Uso e implementación de *MATLAB Visualizations*.

6.2.3.3.4 Alertas.

Una de las cosas importantes en el momento de colectar información de un sistema de monitoreo en tiempo real, es poder actuar sobre ella y así generar algún tipo de acción proactiva, en este apartado se verificará la forma en que se deberán generar alertas basados en el movimiento que tenga el prototipo.

Teniendo en cuenta el auge de los dispositivos móviles y realizando una búsqueda, se halla la herramienta *IoT ThingSpeak Monitor Widget* para equipos con sistema operativo Android, es completamente gratuita. La aplicación tiene las siguientes características:

i. Crear widgets para monitorear los valores reales de los campos de los canales de ThingSpeak, uno o dos en cada artilugio.
ii. Supervisar muchos campos de diferentes canales que crean varios widgets en una pantalla.
iii. Monitorizar los canales privados a través de las llaves de lectura.
iv. Establecer los umbrales de alerta superior e inferior para recibir **alertas**, si el valor del campo programado sobrepasa estos límites.
v. Ver y personalizar gráficos de los valores almacenados.

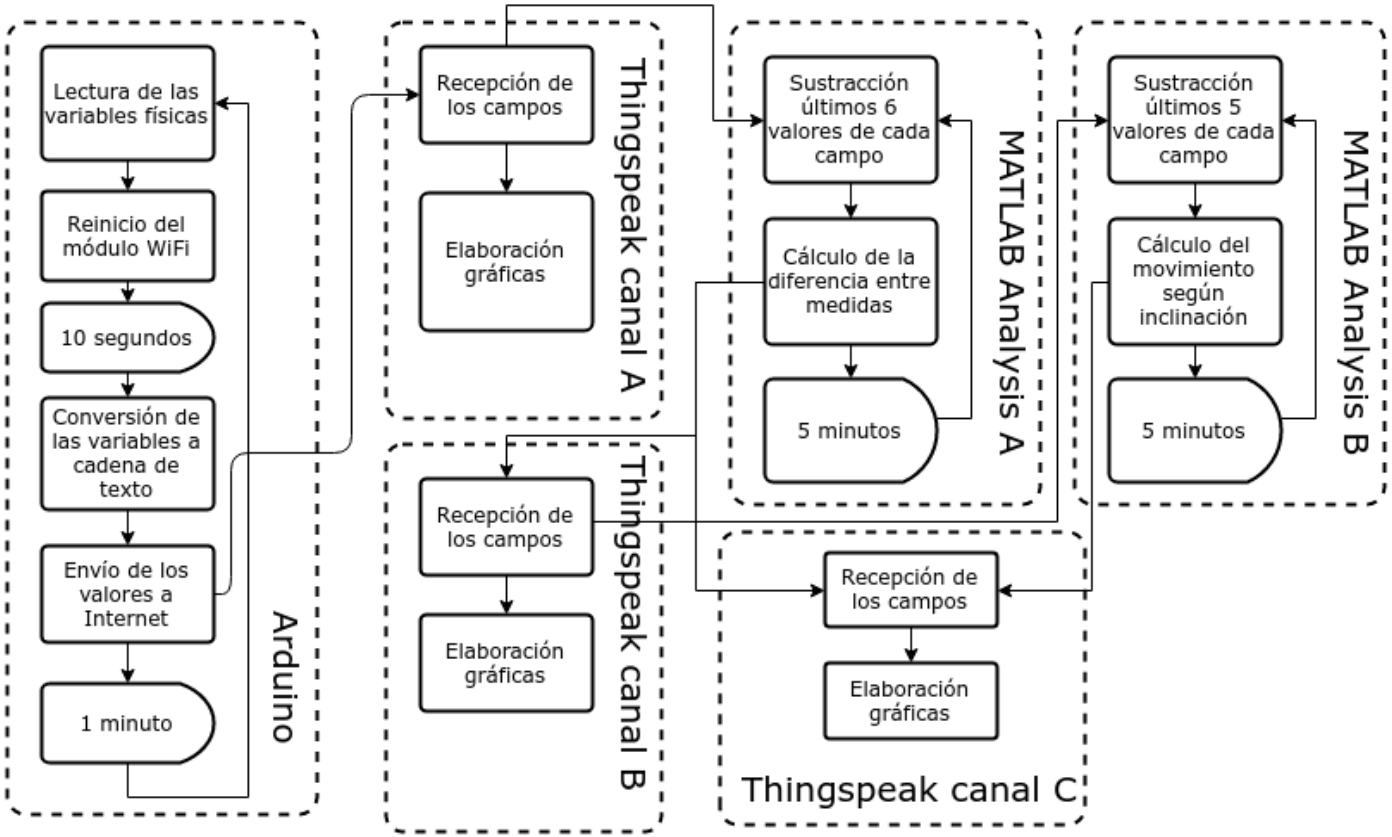
Se instala y se prueba con resultados satisfactorios dadas las características mencionadas previamente.
En la fase de pruebas del modelo a escala, se detallarán los umbrales definidos para las alertas.

El detalle de las pruebas iniciales y configuración de la aplicación *IoT ThingSpeak Monitor Widget* es posible verlo en:
Anexo K. Pruebas de IoT ThingSpeak Monitor Widget.

6.3 DIAGRAMA DEL PROCESO GENERAL DEL PROTOTIPO.

Posterior a la modelación del prototipo y pruebas básicas de operación, a través del esquema de la Figura 63 es posible apreciar el proceso general de envío de información del prototipo, almacenamiento y análisis de la misma.

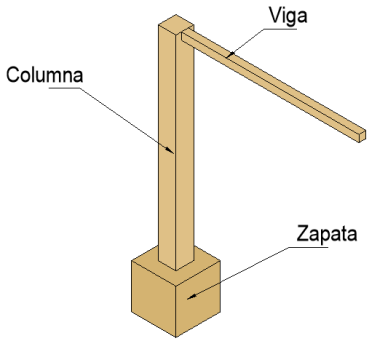
Figura 63. Proceso general del prototipo



7.PRUEBAS CON EL MODELO A ESCALA

Como se indicó en los objetivos específicos, el prototipo diseñado será probado en un modelo a escala, para lo cual se diseñó el modelo básico de un sistema de pórticos, es decir una columna con zapata y una viga, tal y como se muestra en la Figura 64.

Figura 64. Modelo en 3D de la estructura



Dicha estructura modelo se construyó con madera tipo balso y se empotró en una caja de madera con sustancia moldeable para simular los movimientos verticales y horizontales tal y como se muestra en la Figura **65**.

A la estructura a escala se le incorpora el prototipo en la parte superior de la columna, también se toma la decisión de monitorear el movimiento en Z de la viga a través de otro ultrasonido HC-SR04, por ende se modifica el código para la inclusión de este segundo sensor (Anexo F) y se actualiza el diagrama completo de conexiones (Figura **66**). La alimentación del prototipo será realizada con una pila de 9 V. De igual forma se determina que el prototipo envíe la información capturada con una frecuencia de un (1) minuto.
Los planos de diseño de la estructura a escala y adicional los del prototipo son posible apreciarlos en **Anexo L. Planos del diseño de la estructura y prototipo.**

Figura 65. Escenario de prueba

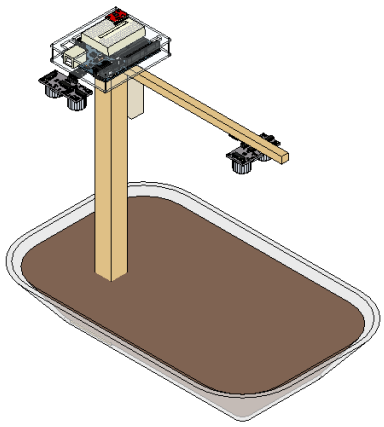
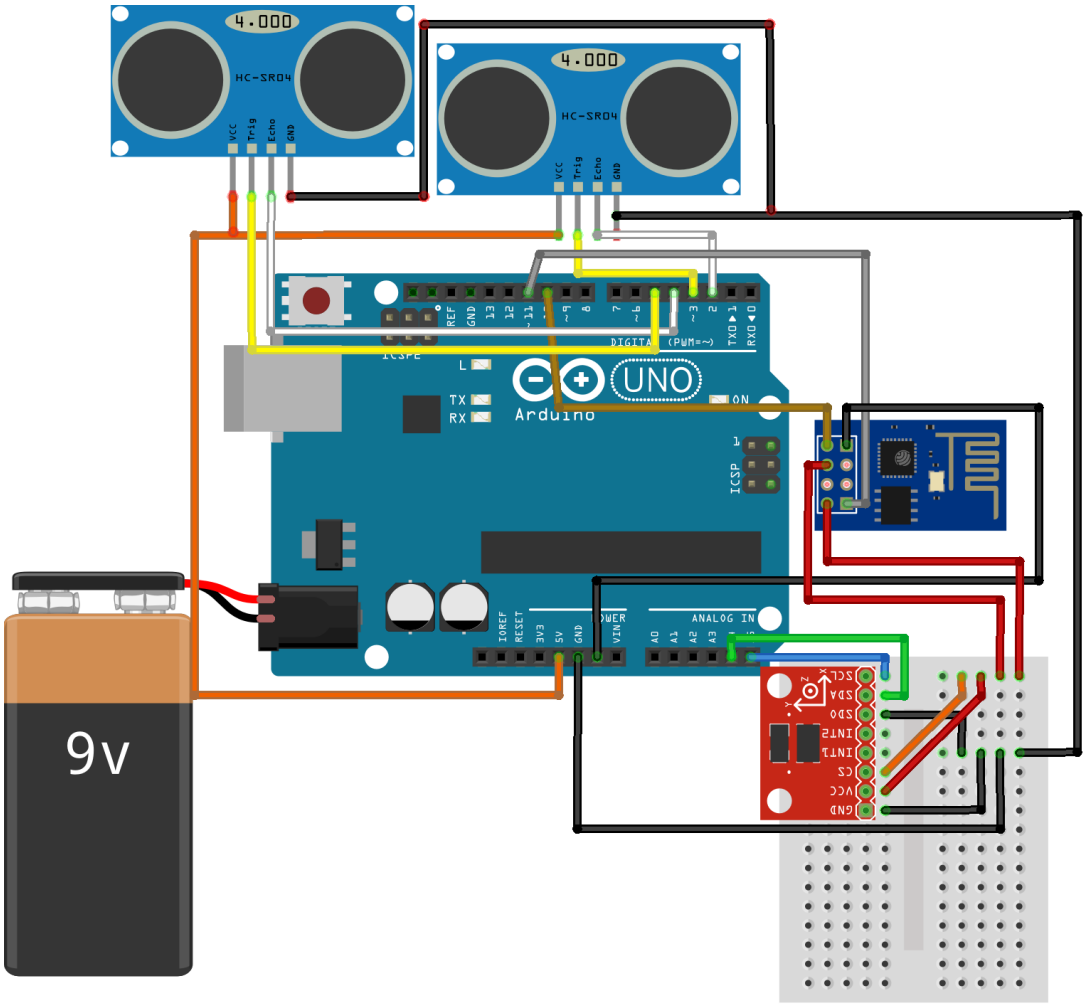


Figura 66. Diagrama completo de conexiones actualizado



7.1 PRUEBAS DEL MOVIMIENTO VERTICAL (ASENTAMIENTO)

7.1.1 Detalles del escenario de pruebas

Como se indicó en la parte introductoria de este apartado, al modelo se le adicionó un sensor adicional para medir el desplazamiento de la viga, por ende, se tienen dos sensores de ultrasonido en el prototipo, uno para la columna y otro para la viga.

7.1.1.1 Escenario 1: Medida de la distancia

Se toman medidas de la distancia física que se encuentra entre los ultrasonidos y el obstáculo que simula el piso donde rebotarían las ondas de sonido, teniendo las siguientes medidas aproximadamente:

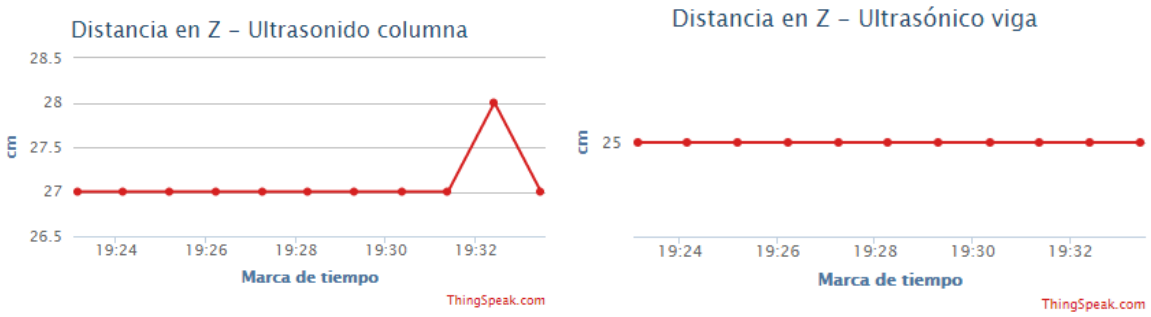
a) Distancia en la columna: 27 cm
b) Distancia en la viga: 25 cm

Las medidas es posible apreciarlas en la

Figura **67**. Además en la figura Figura 68 se visualizan las medidas enviadas y registradas al canal en ThingSpeak, las cuales comprenden a la información tomada por cada ultrasonido.

Figura 67. Medida tomada de la columna y viga

Figura 68. Distancia tomada por el ultrasonido



7.1.1.2 Escenario 2: Medida del asentamiento de la columna y la viga

Se eleva la estructura a escala, cierta distancia dentro de la sustancia moldeable para poder probar que en el momento de simular un asentamiento de la estructura, los valores de la distancia de los ultrasonidos disminuyan. Para llegar a ello, simplemente pondremos

el prototipo sobre la sustancia moldeable, y lo “hundiremos” en el agujero que se logra apreciar en la Figura 69.

Figura 69. Agujero

Se toman de nuevo medidas iniciales con la estructura elevada, teniendo las siguientes medidas aproximadamente:

a) Distancia en la columna: 29 cm
b) Distancia en la viga: 27 cm

Las medidas es posible apreciarlas en la Figura 70.

Figura 70. Medidas escenario 2

7.1.2 Resultado de las pruebas realizadas

7.1.2.1 Resultados escenario 1

Se deja el prototipo enviando los resultados de las medidas por un tiempo de 60 minutos, es decir un total de 60 muestras, se exportan los datos a un archivo separado por comas para poder procesar dicha información a través de Excel obteniendo los siguientes datos estadísticos de las medidas tomadas para cada uno de los dos sensores.

Figura 71. Gráficas de las medidas tomadas

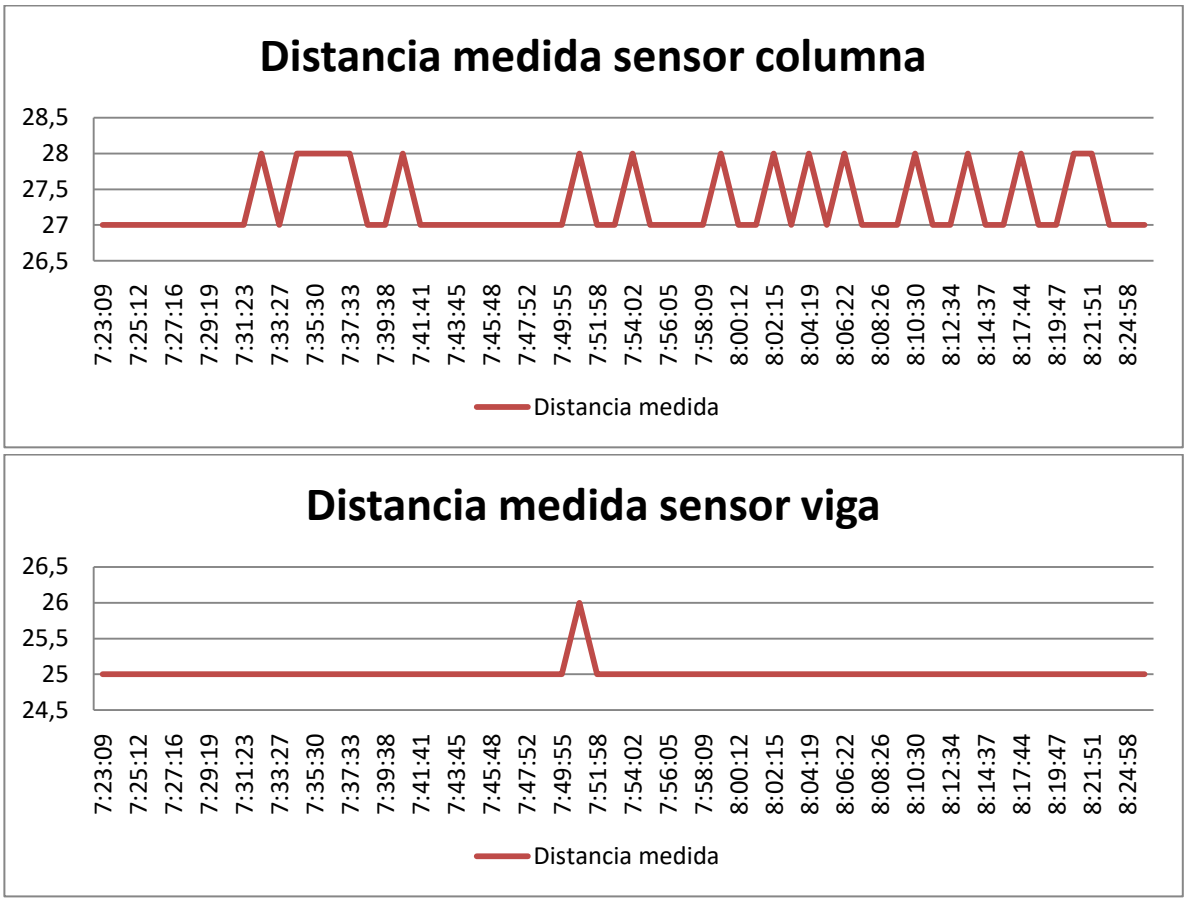


Tabla 16. Datos estadísticos de las medidas escenario 1

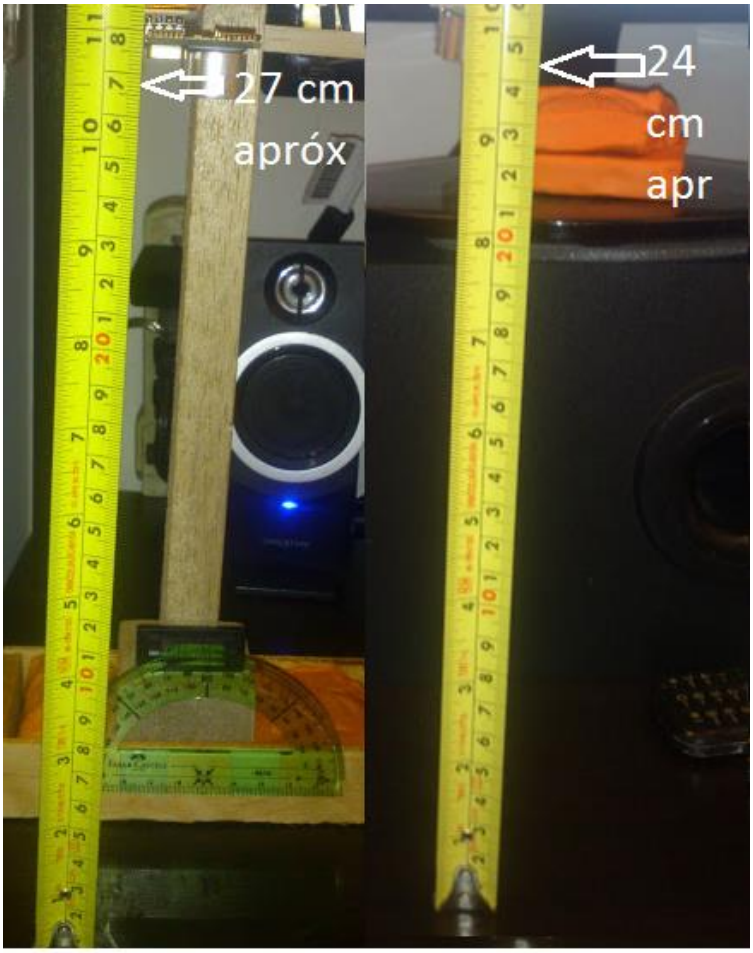
Dato estadístico	Valor sensor columna	Valor sensor viga
Valor real medido	27	25
Cantidad de muestras	60	60
Media valor medido	27,283 cm	25,017 cm
% error según promedio	1,049%	0,067%
% de medidas erradas vs total medidas	28,33%	1,66%

La muestra de las 60 medidas para los dos sensores tomadas es posible apreciarlas en el:
Anexo M: Medidas tomadas ultrasonidos

7.1.2.2 Resultados escenario 2

Como se explicó previamente, se inserta la base de la estructura a escala en el socavón para verificar que tome la variación de la medida, en la Figura 72 se logra apreciar las medidas posteriores a la variación o simulación del asentamiento.

Figura 72. Medidas post asentamiento



Con el fin de corroborar la variación de la medida dada por el asentamiento de la estructura a escala, se verifican las medidas tomadas y registradas por el canal en ThingSpeak, en las siguientes figuras es posible visualizar la variación de las medidas tomadas, así como la cantidad de movimiento calculada a través de la app descrita anteriormente en el apartado 6.2.3.3.1

Figura 73. Variación de la medida - columna

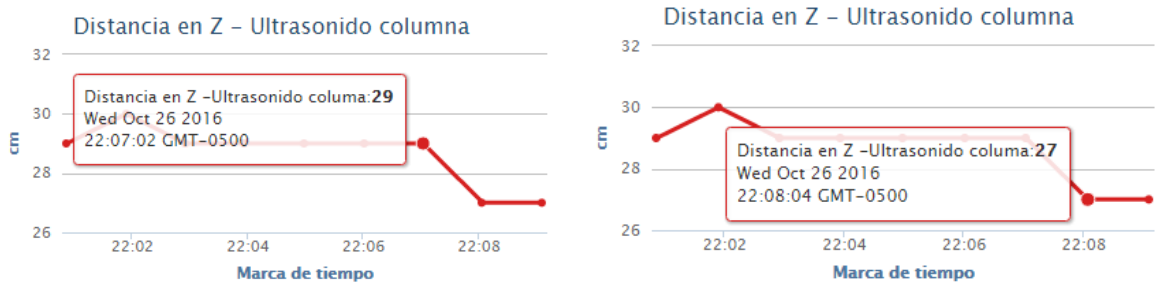


Figura 74. Variación de la medida - viga

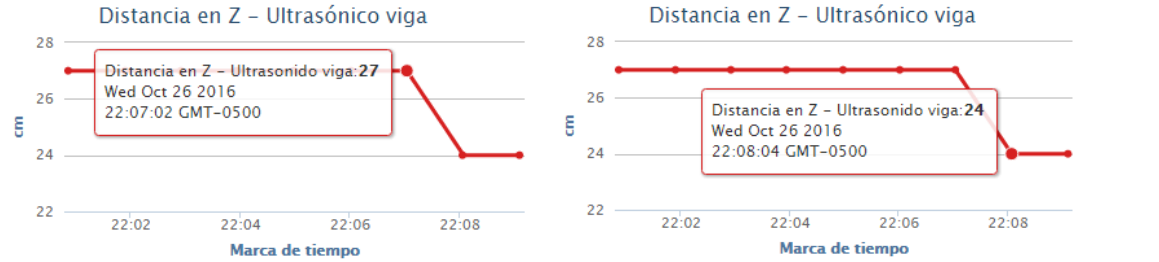


Figura 75. Curvas de la variación de las medidas

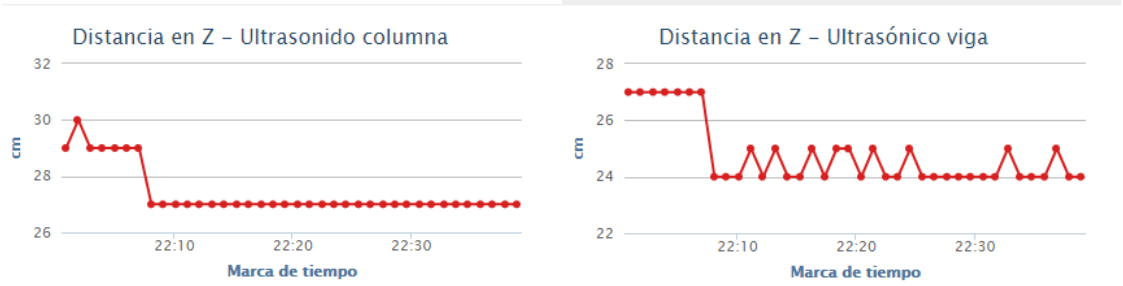


Figura 76. Registro de la cantidad total de movimiento - columna

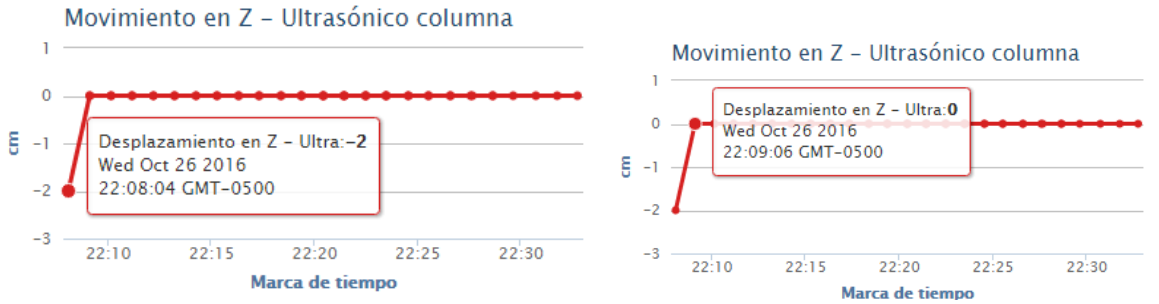
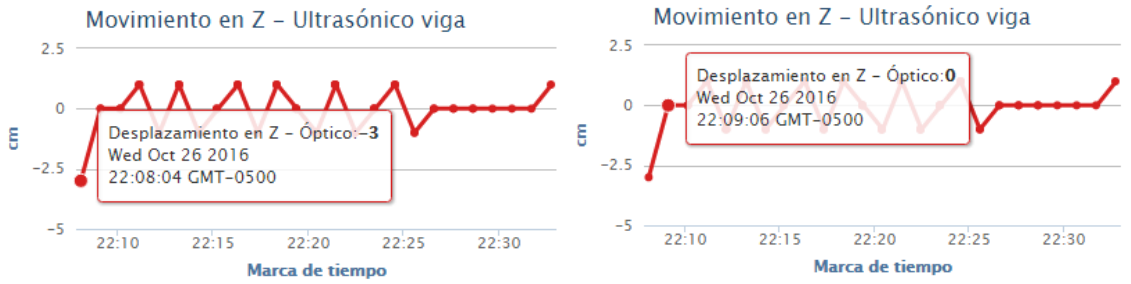


Figura 77. Registro de la cantidad total de movimiento - viga

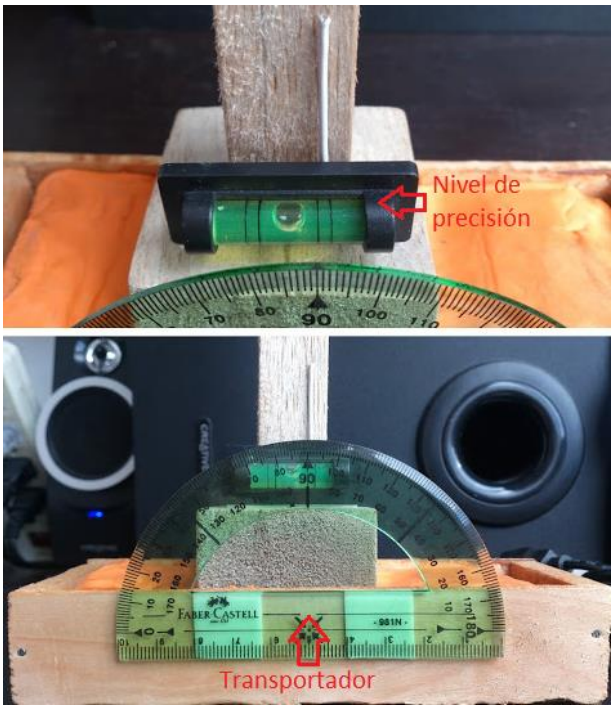


7.2 PRUEBAS DEL MOVIMIENTO HORIZONTAL (DERIVA)

7.2.1 Detalles del escenario de pruebas

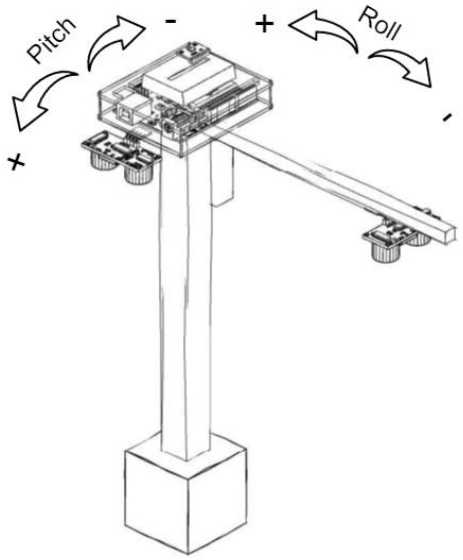
La estructura a escala sigue estando entre la sustancia moldeable, en la base de la misma se situó un nivel de precisión para saber que el modelo a escala se encuentra lo más nivelado posible, adicional, se instaló un transportador para verificar la cantidad de grados en los que se mueve la estructura (Figura 78). Se inclinará la estructura 5, 10, y 15 grados respectivamente para verificar la precisión del prototipo.

Figura 78. Escenario de pruebas Mov. Horizontal



Es importante tener la claridad con el prototipo ya instalado en la estructura a escala, que las medidas en grados se comportarán según lo indicado en la Figura 41 y **¡Error! No se encuentra el origen de la referencia.**. De igual forma en la siguiente figura (Figura 79) se logra ver cuál será el comportamiento de los valores con el prototipo ya situado en la estructura a escala.

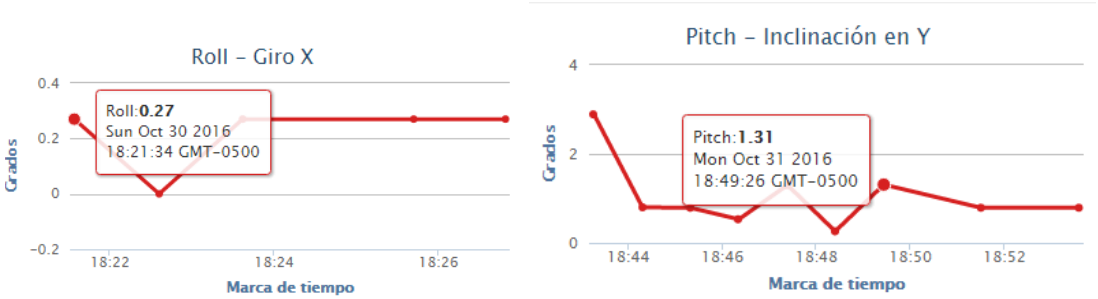
Figura 79.Comportamiento del valor Pitch y Roll en el modelo a escala



7.2.2 Resultado de las pruebas realizadas.

En primera instancia, se garantiza que el prototipo se encuentra lo más estable posible con el nivel de precisión mostrado en la Figura 78. Para lo cual se muestra en las medidas tomadas en el canal de Thingspeak para los valores Pitch y Roll, posterior a la estabilización.

Figura 80. Valores iniciales Roll - Giro en X y Pitch – Inclinación en Y

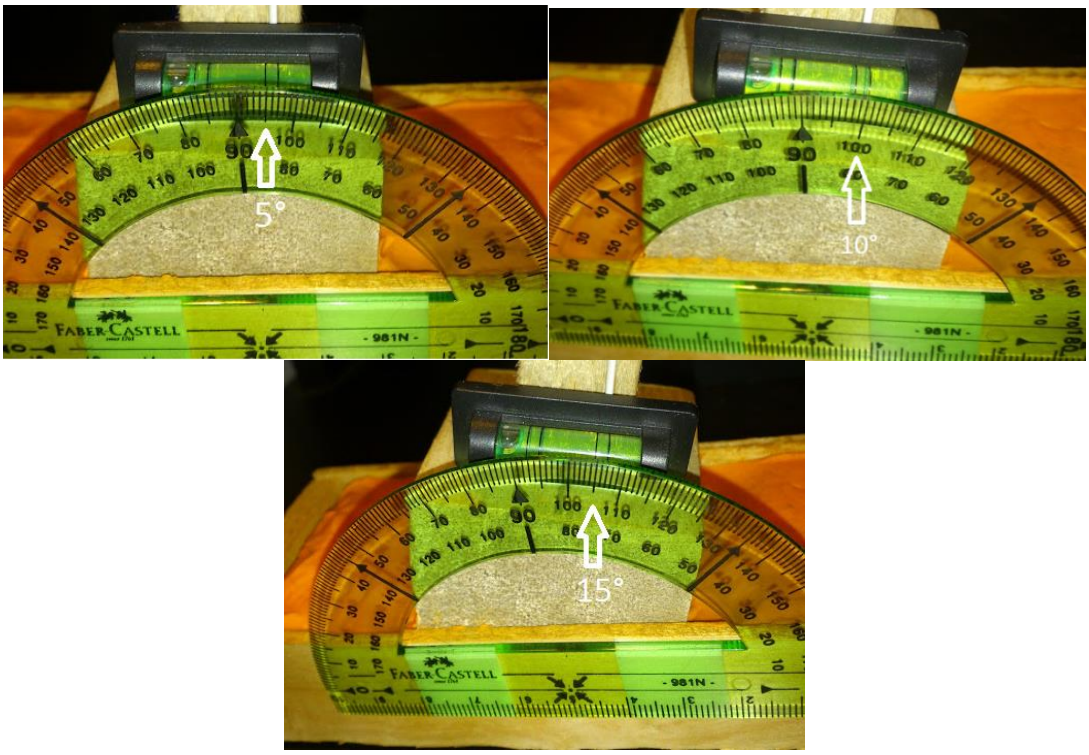


Las medidas respectivamente se encuentran entre los rangos:

a) Roll - Giro en X: {0°, 0.27°}
b) Pitch – Inclinación en Y: {0.8°, 1.31°}

Y como se indicó inicialmente, se inclinó aproximadamente 5° hasta llegar a los 15° tal y como se ve en la Figura **81**.

Figura 81. Registro del movimiento en grados



Es importante tener claro que el movimiento se realiza manualmente, y es posible no tener la medida en grados exacta debido al error humano del movimiento.

7.2.2.1 Resultados de la medida del Roll – Giro en X

Para cada una de las siguientes evidencias, se debe tener en consideración que el prototipo se movió hacia la derecha, es decir que los valores aumentaran en la escala negativa, tal y como se mostró en la Figura 79. También se debe considerar en cada figura la gráfica de la derecha muestra el valor inicial y la de la izquierda el valor posterior al movimiento. La cantidad de movimiento calculada a través de la app descrita anteriormente en el apartado 6.2.3.3.1 se logra apreciar en la

Figura **86**.

Figura 82. Medida al moverlo 5 grados

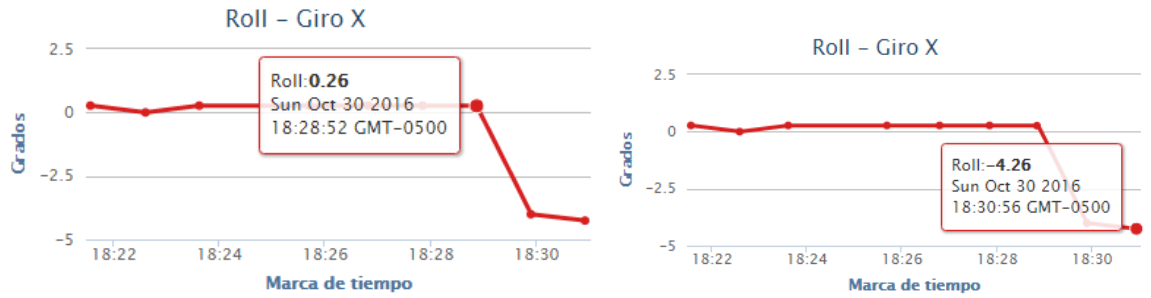


Figura 83. Medida al moverlo 10 grados

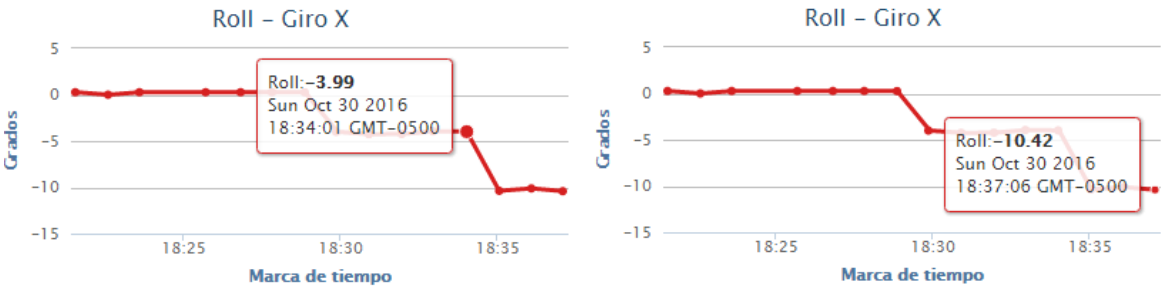


Figura 84. Medida al moverlo 15 grados

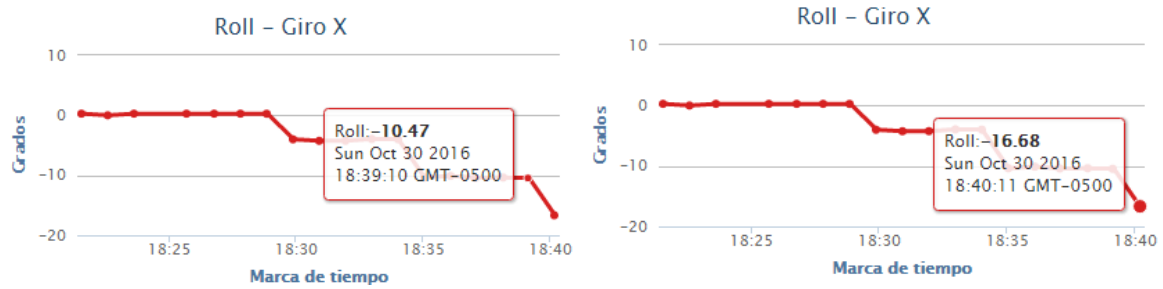


Figura 85. Registro total del movimiento

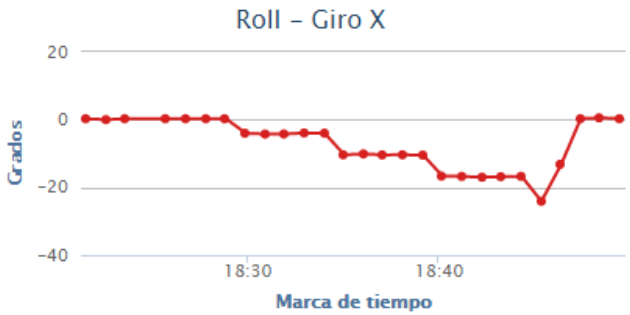
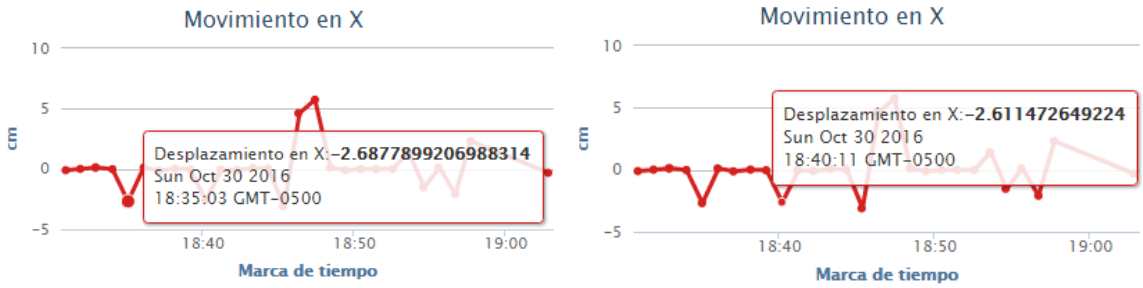


Figura 86. Cantidad de movimiento en cm



Según los descrito en el apartado 4.2.1 la cantidad de movimiento reflejado en la

Figura **86** se calcula basado en la ecuación:

$$tan\theta = \frac{opuesto}{adyacente} \Longrightarrow tang\theta = \frac{\Delta}{h} \Longrightarrow \Delta = tang\theta * h$$

Basados en la cantidad de grados en los que se mueve el prototipo (5°) y la altura (24 cm) de la columna de la estructura a escala, tenemos como valor teórico:

$$\Delta = tan5 * 24\ cm = 2{,}01\ cm$$

La cantidad de movimiento reflejada en

Figura **86** muestra como valores de desplazamiento 2,68 y 2,61 cm, con estas medidas calculamos un error de 33 y 30 % aproximadamente, aunque cabe resaltar que como se mencionó al inicio del apartado: el movimiento se realiza manualmente, y es posible no tener la medida en grados exacta debido al error humano del movimiento.
Lo anterior dado a que se refleja una alta sensibilidad del acelerómetro.

7.2.2.2 Resultados de la medida del Pitch – Inclinación en Y

Figura 87. Medida al moverlo 5 grados

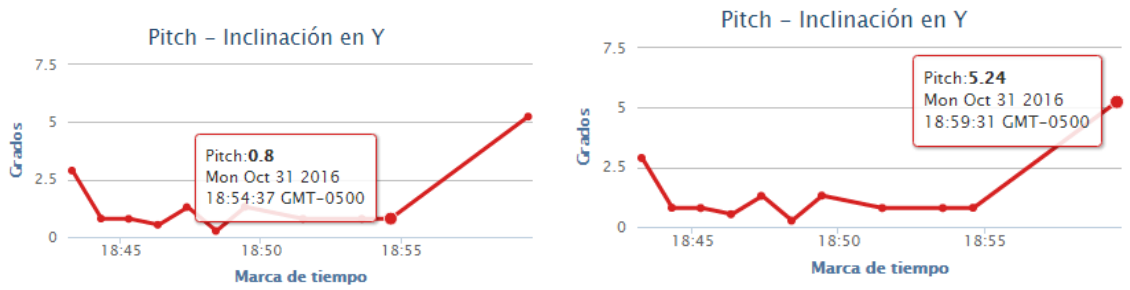
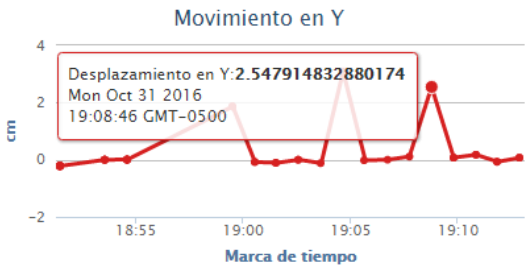


Figura 88. Medida al moverlo 10 grados

Figura 89. Medida al moverlo 15 grados

Figura 90. Registro total de movimiento

Figura 91. Cantidad de movimiento en cm



De igual forma con el movimiento anterior, según los descrito en el apartado 4.2.1 la cantidad de movimiento reflejado en la Figura **91** se calcula basado en la ecuación:

$$tan\theta = \frac{opuesto}{adyacente} \Longrightarrow tang\theta = \frac{\Delta}{h} \Longrightarrow \Delta = tang\theta * h$$

Basados en la cantidad de grados en los que se mueve el prototipo (5°) y la altura (24 cm) de la columna de la estructura a escala, tenemos como valor teórico:

$$\Delta = tan5 * 24\ cm = 2{,}01\ cm$$

La cantidad de movimiento reflejada en Figura **91** muestra como valores de desplazamiento 1,86 y 2,54 cm, con estas medidas calculamos un error de -7 y 26 % aproximadamente, aunque cabe resaltar que como se mencionó al inicio del apartado: el movimiento se realiza manualmente, y es posible no tener la medida en grados exacta debido al error humano del movimiento.
Lo anterior dado a que se refleja una alta sensibilidad del acelerómetro.

7.3 CONCLUSIONES DEL PROCESO DE PRUEBAS

a) En referencia a la medida del asentamiento, se comprueba inestabilidad de los sensores ultrasónicos con ciertos cambios abruptos en las medidas, pero en síntesis no afectarían gravemente la interpretación, dado que lo que se requiere medir es la disminución de la distancia, y los cambios como se puede apreciar en las gráficas de la Figura 71 siempre son en aumento, no en disminución.
b) En referencia a las medidas de los ángulos de inclinación, se comprueba que el acelerómetro seleccionado tiene una precisión bastante alta (menos de 1°), que incluso la intervención humana y manual del movimiento genera errores considerables.

8.RECOMENDACIONES ADICIONALES AL PROTOTIPO

Basado en el proceso de experimentación desarrollado, se lista a continuación una serie de mejoras para desarrollar a futuro, con el fin de tener un prototipo mucho más robusto.

a) Precisión de la medida del asentamiento: debido a la naturaleza del método de medición de los sensores ultrasónicos, están propensos a sufrir errores en el cálculo de la distancia debido a factores tales como obstáculos y otros como la velocidad en que las ondas se propagan en el aire, dado que factores tales como la temperatura afectan a las mismas. Para mitigar dichos errores, se propone que se utilice un módulo telémetro láser como el descrito en el apartado 5.1.2.3, que lastimosamente por el costo no fue posible adquirir para este prototipo.
b) Consumo de energía: se evidencia que el prototipo puede llegar a consumir una batería de 9 V en aproximadamente 8 horas, para lo cual se propone que el mismo no envíe información en un lapso tan corto de tiempo (1 min) dado que en movimiento de las estructuras no requieren un monitoreo tan detallado. También se propone que el dispositivo entre en un modo de bajo consumo de energía a través de librerías de Arduino tales como ArduinoSleepCode[99], que permiten que el dispositivo hiberne y a través de un disparador se active para tomar la medida y vuelva al estado de hibernación.
c) Respaldo de la información almacenada en los canales de ThingSpeak: Debido a que la plataforma de almacenamiento de información es de código abierto, se recomienda a través de las llaves de lectura y lecturas via web, generar una tarea automática que realice un respaldo de la información almacenada.
d) Tamaño del prototipo: es posible a través de plataformas como Arduino Pro Mini, Arduino Pro MICRO o Arduino MKR1000 generar un prototipo mucho más pequeño.

[99] http://playground.arduino.cc/Learning/ArduinoSleepCode

9.CONCLUSIONES GENERALES DEL PROYECTO

- A partir de la experimentación, se comprueba la versatilidad de plataformas como Arduino para el desarrollo de prototipos, lo anterior dado el tiempo que lleva en el mercado, la amplia compatibilidad de dispositivos y la gran comunidad que empíricamente enriquece a la misma.

- En el proceso de desarrollo e investigación, se encuentran herramientas como Thingspeak, que simplifican la forma en que almacena, visualiza y manipula la información obtenida por los sensores, y lo mejor de todo, está orientada a la nube: puede ser consultada en cualquier parte del mundo a través de internet.

- Los errores de precisión siempre serán un punto de mejora en el desarrollo de prototipos.

- El internet de las cosas, actualmente, es un campo de gran emprendimiento. El módulo WIFI ESP 8266 dada su economía y versatilidad se orienta como un gran dispositivo para este campo.

- El desarrollo y culminación del prototipo abre el horizonte hacia nuevas ideas y desarrollos personales.

- Es gratificante cumplir los objetivos trazados y que el desafío inicial del diseño, desarrollo e implementación sienten las bases para un conocimiento mucho más amplio.

BIBLIOGRAFÍA

Addleson, Lyall. 1983. *Materiales para la construcción, Volumen 1.* s.l. : Reverte, 1983.

Arduino. 2014. Arduino playground. [En línea] 2014. http://playground.arduino.cc/.

Awad, Roberto Rochel. 2012. Análisis y diseño sísmico de edificios. Medellín: Fondo editorial Universidad EAFIT, 2012.

Comisión Asesora Permanente del Régimen de Construcciones Sismo Resistentes. 2010. NORMAS SISMO RESISTENTES COLOMBIANAS. 2010, CAPÍTULO A.1 , pág. 1.

Creaciones. 2010. *Guía práctica de sensores.* s.l. : Creaciones, 2010.

Española, Real Academia de la lengua. 2001. *Diccionario de la lengua española.* 2001.

José Manuel Huidobro, Ramón Jesús Millán Tejedor. 2010. *Manual de domótica.* s.l. : Creaciones, 2010.

Lex Nova. 2009. *Higiene industrial 9.a ed.* s.l. : Lex Nova, 2009. pág. 366.

Santamaría, Eduardo. 1993. Electrónica digital y microprocesadores. Univ Pontifica Comillas. pág. 257.

Caicedo Pedrera, Antonio. 2014. Arduino para Principiantes. IT Campus Academy. pág. 10.

Mosquera, Genaro. 2000. Las vibraciones mecánicas y su aplicación al mantenimiento predictivo. pág. 111.

Pallás Areny, Ramón. 2004. Sensores y acondicionadores de señal. Marcombo. pág. 4.

Maloney, Timothy J. 2006. Electrónica industrial moderna. Pearson Education. pág. 439.

Reverte. 2002. Los sensores en el automóvil. Bosch. pág. 37.

Menéndez, Faustino. 2009. Higiene industrial 9.a ed. Lex Nova. pág. 366.

Anexo A. Datasheet acelerómetro ADXL345

3-Axis, ±2 *g*/±4 *g*/±8 *g*/±16 *g* Digital Accelerometer

ADXL345

FEATURES

- Ultralow power: as low as 40 µA in measurement mode and 0.1 µA in standby mode at V_S = 2.5 V (typical)
- Power consumption scales automatically with bandwidth
- User-selectable resolution
  - Fixed 10-bit resolution
  - Full resolution, where resolution increases with *g* range, up to 13-bit resolution at ±16 *g* (maintaining 4 mg/LSB scale factor in all *g* ranges)
- Embedded, patent pending FIFO technology minimizes host processor load
- Tap/double tap detection
- Activity/inactivity monitoring
- Free-fall detection
- Supply voltage range: 2.0 V to 3.6 V
- I/O voltage range: 1.7 V to V_S
- SPI (3- and 4-wire) and I^2C digital interfaces
- Flexible interrupt modes mappable to either interrupt pin
- Measurement ranges selectable via serial command
- Bandwidth selectable via serial command
- Wide temperature range (−40°C to +85°C)
- 10,000 *g* shock survival
- Pb free/RoHS compliant
- Small and thin: 3 mm × 5 mm × 1 mm LGA package

APPLICATIONS

- Handsets
- Medical instrumentation
- Gaming and pointing devices
- Industrial instrumentation
- Personal navigation devices
- Hard disk drive (HDD) protection
- Fitness equipment

GENERAL DESCRIPTION

The ADXL345 is a small, thin, low power, 3-axis accelerometer with high resolution (13-bit) measurement at up to ±16 *g*. Digital output data is formatted as 16-bit twos complement and is accessible through either a SPI (3- or 4-wire) or I^2C digital interface.

The ADXL345 is well suited for mobile device applications. It measures the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion or shock. Its high resolution (4 mg/LSB) enables measurement of inclination changes less than 1.0°.

Several special sensing functions are provided. Activity and inactivity sensing detect the presence or lack of motion and if the acceleration on any axis exceeds a user-set level. Tap sensing detects single and double taps. Free-fall sensing detects if the device is falling. These functions can be mapped to one of two interrupt output pins. An integrated, patent pending 32-level first in, first out (FIFO) buffer can be used to store data to minimize host processor intervention.

Low power modes enable intelligent motion-based power management with threshold sensing and active acceleration measurement at extremely low power dissipation.

The ADXL345 is supplied in a small, thin, 3 mm × 5 mm × 1 mm, 14-lead, plastic package.

FUNCTIONAL BLOCK DIAGRAM

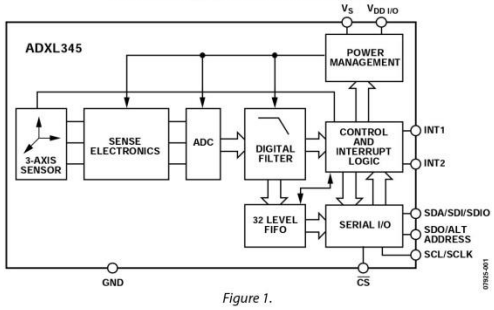


Figure 1.

Rev. 0

Information furnished by Analog Devices is believed to be accurate and reliable. However, no responsibility is assumed by Analog Devices for its use, nor for any infringements of patents or other rights of third parties that may result from its use. Specifications subject to change without notice. No license is granted by implication or otherwise under any patent or patent rights of Analog Devices. Trademarks and registered trademarks are the property of their respective owners. See the last page for disclaimers.

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700 www.analog.com
Fax: 781.461.3113

©2009 Analog Devices, Inc. All rights reserved.

TABLE OF CONTENTS

REVISION HISTORY

5/09—Revision 0: Initial Version

SPECIFICATIONS

T_A = 25°C, V_S = 2.5 V, $V_{DD\ I/O}$ = 1.8 V, acceleration = 0 *g*, C_S = 1 µF tantalum, C_{IO} = 0.1 µF, unless otherwise noted.

Table 1. Specifications[1]

Parameter	Test Conditions	Min	Typ	Max	Unit
SENSOR INPUT	Each axis				
Measurement Range	User selectable		±2, ±4, ±8, ±16		*g*
Nonlinearity	Percentage of full scale		±0.5		%
Inter-Axis Alignment Error			±0.1		Degrees
Cross-Axis Sensitivity[2]			±1		%
OUTPUT RESOLUTION	Each axis				
All *g* Ranges	10-bit resolution		10		Bits
±2 *g* Range	Full resolution		10		Bits
±4 *g* Range	Full resolution		11		Bits
±8 *g* Range	Full resolution		12		Bits
±16 *g* Range	Full resolution		13		Bits
SENSITIVITY	Each axis				
Sensitivity at X_{OUT}, Y_{OUT}, Z_{OUT}	±2 *g*, 10-bit or full resolution	232	256	286	LSB/*g*
Scale Factor at X_{OUT}, Y_{OUT}, Z_{OUT}	±2 *g*, 10-bit or full resolution	3.5	3.9	4.3	mg/LSB
Sensitivity at X_{OUT}, Y_{OUT}, Z_{OUT}	±4 *g*, 10-bit resolution	116	128	143	LSB/*g*
Scale Factor at X_{OUT}, Y_{OUT}, Z_{OUT}	±4 *g*, 10-bit resolution	7.0	7.8	8.6	mg/LSB
Sensitivity at X_{OUT}, Y_{OUT}, Z_{OUT}	±8 *g*, 10-bit resolution	58	64	71	LSB/*g*
Scale Factor at X_{OUT}, Y_{OUT}, Z_{OUT}	±8 *g*, 10-bit resolution	14.0	15.6	17.2	mg/LSB
Sensitivity at X_{OUT}, Y_{OUT}, Z_{OUT}	±16 *g*, 10-bit resolution	29	32	36	LSB/*g*
Scale Factor at X_{OUT}, Y_{OUT}, Z_{OUT}	±16 *g*, 10-bit resolution	28.1	31.2	34.3	mg/LSB
Sensitivity Change Due to Temperature			±0.01		%/°C
0 *g* BIAS LEVEL	Each axis				
0 *g* Output for X_{OUT}, Y_{OUT}		−150	±40	+150	m*g*
0 *g* Output for Z_{OUT}		−250	±80	+250	m*g*
0 *g* Offset vs. Temperature for x-, y-Axes			±0.8		mg/°C
0 *g* Offset vs. Temperature for z-Axis			±4.5		mg/°C
NOISE PERFORMANCE					
Noise (x-, y-Axes)	Data rate = 100 Hz for ±2 *g*, 10-bit or full resolution		<1.0		LSB rms
Noise (z-Axis)	Data rate = 100 Hz for ±2 *g*, 10-bit or full resolution		<1.5		LSB rms
OUTPUT DATA RATE AND BANDWIDTH	User selectable				
Measurement Rate[3]		6.25		3200	Hz
SELF-TEST[4]	Data rate ≥ 100 Hz, 2.0 V ≤ V_S ≤ 3.6 V				
Output Change in x-Axis		0.20		2.10	*g*
Output Change in y-Axis		−2.10		−0.20	*g*
Output Change in z-Axis		0.30		3.40	*g*
POWER SUPPLY					
Operating Voltage Range (V_S)		2.0	2.5	3.6	V
Interface Voltage Range ($V_{DD\ I/O}$)	V_S ≤ 2.5 V	1.7	1.8	V_S	V
	V_S ≥ 2.5 V	2.0	2.5	V_S	V
Supply Current	Data rate > 100 Hz		145		µA
	Data rate < 10 Hz		40		µA
Standby Mode Leakage Current			0.1	2	µA
Turn-On Time[5]	Data rate = 3200 Hz		1.4		ms
TEMPERATURE					
Operating Temperature Range		−40		+85	°C
WEIGHT					
Device Weight			20		mg

[1] All minimum and maximum specifications are guaranteed. Typical specifications are not guaranteed.
[2] Cross-axis sensitivity is defined as coupling between any two axes.
[3] Bandwidth is half the output data rate.
[4] Self-test change is defined as the output (*g*) when the SELF_TEST bit = 1 (in the DATA_FORMAT register) minus the output (*g*) when the SELF_TEST bit = 0 (in the DATA_FORMAT register). Due to device filtering, the output reaches its final value after 4 × τ when enabling or disabling self-test, where τ = 1/(data rate).
[5] Turn-on and wake-up times are determined by the user-defined bandwidth. At a 100 Hz data rate, the turn-on and wake-up times are each approximately 11.1 ms. For other data rates, the turn-on and wake-up times are each approximately τ + 1.1 in milliseconds, where τ = 1/(data rate).

ABSOLUTE MAXIMUM RATINGS

Table 2.

Parameter	Rating
Acceleration	
Any Axis, Unpowered	10,000 *g*
Any Axis, Powered	10,000 *g*
V_S	−0.3 V to +3.6 V
$V_{DD\ I/O}$	−0.3 V to +3.6 V
Digital Pins	−0.3 V to $V_{DD\ I/O}$ + 0.3 V or 3.6 V, whichever is less
All Other Pins	−0.3 V to +3.6 V
Output Short-Circuit Duration (Any Pin to Ground)	Indefinite
Temperature Range	
Powered	−40°C to +105°C
Storage	−40°C to +105°C

Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; functional operation of the device at these or any other conditions above those indicated in the operational section of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

THERMAL RESISTANCE

Table 3. Package Characteristics

Package Type	θ_{JA}	θ_{JC}	Device Weight
14-Terminal LGA	150°C/W	85°C/W	20 mg

ESD CAUTION

ESD (electrostatic discharge) sensitive device. Charged devices and circuit boards can discharge without detection. Although this product features patented or proprietary protection circuitry, damage may occur on devices subjected to high energy ESD. Therefore, proper ESD precautions should be taken to avoid performance degradation or loss of functionality.

PIN CONFIGURATION AND FUNCTION DESCRIPTIONS

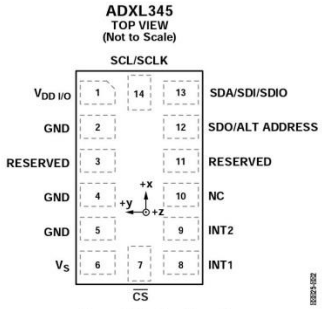


Figure 2. Pin Configuration

Table 4. Pin Function Descriptions

Pin No.	Mnemonic	Description
1	$V_{DD\ I/O}$	Digital Interface Supply Voltage.
2	GND	Must be connected to ground.
3	Reserved	Reserved. This pin must be connected to V_S or left open.
4	GND	Must be connected to ground.
5	GND	Must be connected to ground.
6	V_S	Supply Voltage.
7	$\overline{CS}$	Chip Select.
8	INT1	Interrupt 1 Output.
9	INT2	Interrupt 2 Output.
10	NC	Not Internally Connected.
11	Reserved	Reserved. This pin must be connected to ground or left open.
12	SDO/ALT ADDRESS	Serial Data Output/Alternate I^2C Address Select.
13	SDA/SDI/SDIO	Serial Data (I^2C)/Serial Data Input (SPI 4-Wire)/Serial Data Input and Output (SPI 3-Wire).
14	SCL/SCLK	Serial Communications Clock.

THEORY OF OPERATION

The ADXL345 is a complete 3-axis acceleration measurement system with a selectable measurement range of ±2 *g*, ±4 *g*, ±8 *g*, or ±16 *g*. It measures both dynamic acceleration resulting from motion or shock and static acceleration, such as gravity, which allows the device to be used as a tilt sensor.

The sensor is a polysilicon surface-micromachined structure built on top of a silicon wafer. Polysilicon springs suspend the structure over the surface of the wafer and provide a resistance against acceleration forces.

Deflection of the structure is measured using differential capacitors that consist of independent fixed plates and plates attached to the moving mass. Acceleration deflects the beam and unbalances the differential capacitor, resulting in a sensor output whose amplitude is proportional to acceleration. Phase-sensitive demodulation is used to determine the magnitude and polarity of the acceleration.

POWER SEQUENCING

Power can be applied to V_S or $V_{DD\ I/O}$ in any sequence without damaging the ADXL345. All possible power-on modes are summarized in Table 5. The interface voltage level is set with the interface supply voltage, $V_{DD\ I/O}$, which must be present to ensure that the ADXL345 does not create a conflict on the communication bus. For single-supply operation, $V_{DD\ I/O}$ can be the same as the main supply, V_S. In a dual-supply application, however, $V_{DD\ I/O}$ can differ from V_S to accommodate the desired interface voltage, as long as V_S is greater than $V_{DD\ I/O}$.

After V_S is applied, the device enters standby mode, where power consumption is minimized and the device waits for $V_{DD\ I/O}$ to be applied and for the command to enter measurement mode to be received. (This command can be initiated by setting the measure bit in the POWER_CTL register (Address 0x2D).) In addition, any register can be written to or read from to configure the part while the device is in standby mode. It is recommended to configure the device in standby mode and then to enable measurement mode. Clearing the measure bit returns the device to the standby mode.

Table 5. Power Sequencing

Condition	V_S	$V_{DD\ I/O}$	Description
Power Off	Off	Off	The device is completely off, but there is a potential for a communication bus conflict.
Bus Disabled	On	Off	The device is on in standby mode, but communication is unavailable and will create a conflict on the communication bus. The duration of this state should be minimized during power-up to prevent a conflict.
Bus Enabled	Off	On	No functions are available, but the device will not create a conflict on the communication bus.
Standby or Measurement	On	On	At power-up, the device is in standby mode, awaiting a command to enter measurement mode, and all sensor functions are off. After the device is instructed to enter measurement mode, all sensor functions are available.

POWER SAVINGS

Power Modes

The ADXL345 automatically modulates its power consumption in proportion to its output data rate, as outlined in Table 6. If additional power savings is desired, a lower power mode is available. In this mode, the internal sampling rate is reduced, allowing for power savings in the 12.5 Hz to 400 Hz data rate range but at the expense of slightly greater noise. To enter lower power mode, set the LOW_POWER bit (Bit 4) in the BW_RATE register (Address 0x2C). The current consumption in low power mode is shown in Table 7 for cases where there is an advantage for using low power mode. The current consumption values shown in Table 6 and Table 7 are for a V_S of 2.5 V. Current scales linearly with V_S.

Table 6. Current Consumption vs. Data Rate
(T_A = 25°C, V_S = 2.5 V, $V_{DD\ I/O}$ = 1.8 V)

Output Data Rate (Hz)	Bandwidth (Hz)	Rate Code	I_{DD} (μA)
3200	1600	1111	145
1600	800	1110	100
800	400	1101	145
400	200	1100	145
200	100	1011	145
100	50	1010	145
50	25	1001	100
25	12.5	1000	65
12.5	6.25	0111	55
6.25	3.125	0110	40

Table 7. Current Consumption vs. Data Rate, Low Power Mode
(T_A = 25°C, V_S = 2.5 V, $V_{DD\ I/O}$ = 1.8 V)

Output Data Rate (Hz)	Bandwidth (Hz)	Rate Code	I_{DD} (μA)
400	200	1100	100
200	100	1011	65
100	50	1010	55
50	25	1001	50
25	12.5	1000	40
12.5	6.25	0111	40

Auto Sleep Mode

Additional power can be saved if the ADXL345 automatically switches to sleep mode during periods of inactivity. To enable this feature, set the THRESH_INACT register (Address 0x25) and the TIME_INACT register (Address 0x26) each to a value that signifies inactivity (the appropriate value depends on the application), and then set the AUTO_SLEEP bit and the link bit in the POWER_CTL register (Address 0x2D). Current consumption at the sub-8 Hz data rates used in this mode is typically 40 μA for a V_S of 2.5 V.

Standby Mode

For even lower power operation, standby mode can be used. In standby mode, current consumption is reduced to 0.1 μA (typical). In this mode, no measurements are made. Standby mode is entered by clearing the measure bit (Bit 3) in the POWER_CTL register (Address 0x2D). Placing the device into standby mode preserves the contents of FIFO.

SERIAL COMMUNICATIONS

I²C and SPI digital communications are available. In both cases, the ADXL345 operates as a slave. I²C mode is enabled if the $\overline{CS}$ pin is tied high to $V_{DD\ I/O}$. The $\overline{CS}$ pin should always be tied high to $V_{DD\ I/O}$ or be driven by an external controller because there is no default mode if the $\overline{CS}$ pin is left unconnected. Therefore, not taking these precautions may result in an inability to communicate with the part. In SPI mode, the $\overline{CS}$ pin is controlled by the bus master. In both SPI and I²C modes of operation, data transmitted from the ADXL345 to the master device should be ignored during writes to the ADXL345.

SPI

For SPI, either 3- or 4-wire configuration is possible, as shown in the connection diagrams in Figure 3 and Figure 4. Clearing the SPI bit in the DATA_FORMAT register (Address 0x31) selects 4-wire mode, whereas setting the SPI bit selects 3-wire mode. The maximum SPI clock speed is 5 MHz with 100 pF maximum loading, and the timing scheme follows clock polarity (CPOL) = 1 and clock phase (CPHA) = 1.

$\overline{CS}$ is the serial port enable line and is controlled by the SPI master. This line must go low at the start of a transmission and high at the end of a transmission, as shown in Figure 5. SCLK is the serial port clock and is supplied by the SPI master. It is stopped high when $\overline{CS}$ is high during a period of no transmission. SDI and SDO are the serial data input and output, respectively. Data should be sampled at the rising edge of SCLK.

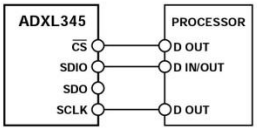


Figure 3. 3-Wire SPI Connection Diagram

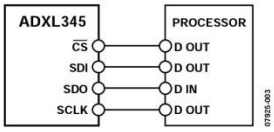


Figure 4. 4-Wire SPI Connection Diagram

To read or write multiple bytes in a single transmission, the multiple-byte bit, located after the R/$\overline{W}$ bit in the first byte transfer (MB in Figure 5 to Figure 7), must be set. After the register addressing and the first byte of data, each subsequent set of clock pulses (eight clock pulses) causes the ADXL345 to point to the next register for a read or write. This shifting continues until the clock pulses cease and $\overline{CS}$ is deasserted. To perform reads or writes on different, nonsequential registers, $\overline{CS}$ must be deasserted between transmissions and the new register must be addressed separately.

The timing diagram for 3-wire SPI reads or writes is shown in Figure 7. The 4-wire equivalents for SPI writes and reads are shown in Figure 5 and Figure 6, respectively.

Table 8. SPI Digital Input/Output Voltage

Parameter	Limit[1]	Unit
Digital Input Voltage		
Low Level Input Voltage (V_{IL})	$0.2 \times V_{DD\ I/O}$	V max
High Level Input Voltage (V_{IH})	$0.8 \times V_{DD\ I/O}$	V min
Digital Output Voltage		
Low Level Output Voltage (V_{OL})	$0.15 \times V_{DD\ I/O}$	V max
High Level Output Voltage (V_{OH})	$0.85 \times V_{DD\ I/O}$	V min

[1] Limits based on characterization results, not production tested.

Table 9. SPI Timing (T_A = 25°C, V_S = 2.5 V, $V_{DD\ I/O}$ = 1.8 V)[1]

	Limit[2, 3]			
Parameter	Min	Max	Unit	Description
f_{SCLK}		5	MHz	SPI clock frequency
t_{SCLK}	200		ns	1/(SPI clock frequency) mark-space ratio for the SCLK input is 40/60 to 60/40
t_{DELAY}	10		ns	$\overline{CS}$ falling edge to SCLK falling edge
t_{QUIET}	10		ns	SCLK rising edge to $\overline{CS}$ rising edge
t_{DIS}		100	ns	$\overline{CS}$ rising edge to SDO disabled
$t_{CS,DIS}$	250		ns	$\overline{CS}$ deassertion between SPI communications
t_S	$0.4 \times t_{SCLK}$		ns	SCLK low pulse width (space)
t_M	$0.4 \times t_{SCLK}$		ns	SCLK high pulse width (mark)
t_{SDO}		95	ns	SCLK falling edge to SDO transition
t_{SETUP}	10		ns	SDI valid before SCLK rising edge
t_{HOLD}	10		ns	SDI valid after SCLK rising edge

[1] The $\overline{CS}$, SCLK, SDI, and SDO pins are not internally pulled up or down; they must be driven for proper operation.
[2] Limits based on characterization results, characterized with f_{SCLK} = 5 MHz and bus load capacitance of 100 pF; not production tested.
[3] The timing values are measured corresponding to the input thresholds (V_{IL} and V_{IH}) given in Table 8.

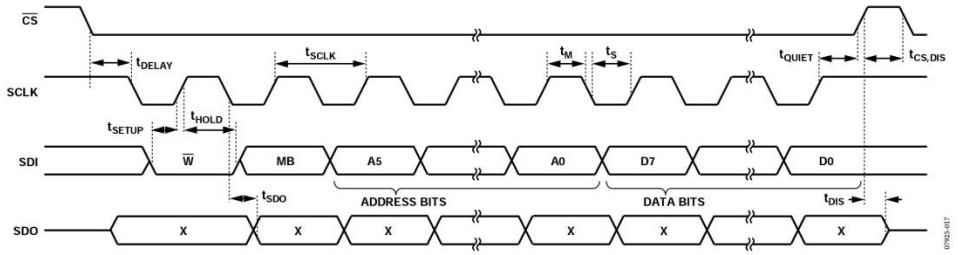


Figure 5. SPI 4-Wire Write

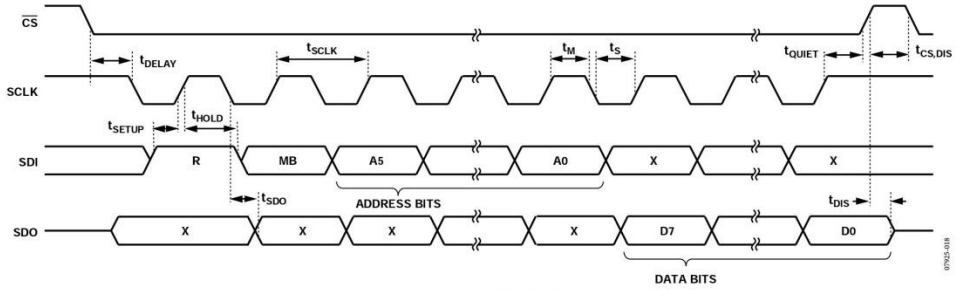


Figure 6. SPI 4-Wire Read

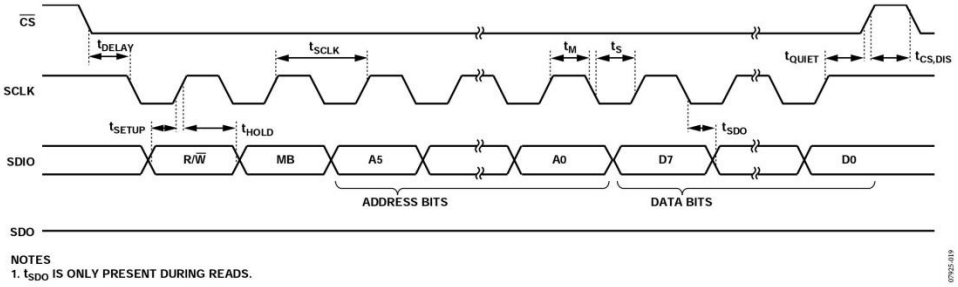


Figure 7. SPI 3-Wire Read/Write

I²C

With $\overline{CS}$ tied high to $V_{DD\ I/O}$, the ADXL345 is in I²C mode, requiring a simple 2-wire connection as shown in Figure 8. The ADXL345 conforms to the *UM10204 I²C-Bus Specification and User Manual*, Rev. 03—19 June 2007, available from NXP Semiconductor. It supports standard (100 kHz) and fast (400 kHz) data transfer modes if the timing parameters given in Table 11 and Figure 10 are met. Single- or multiple-byte reads/writes are supported, as shown in Figure 9. With the SDO/ALT ADDRESS pin high, the 7-bit I²C address for the device is 0x1D, followed by the R/$\overline{W}$ bit. This translates to 0x3A for a write and 0x3B for a read. An alternate I²C address of 0x53 (followed by the R/$\overline{W}$ bit) can be chosen by grounding the SDO/ALT ADDRESS pin (Pin 12). This translates to 0xA6 for a write and 0xA7 for a read.

If other devices are connected to the same I²C bus, the nominal operating voltage level of these other devices cannot exceed $V_{DD\ I/O}$ by more than 0.3 V. External pull-up resistors, R_P, are necessary for proper I²C operation. Refer to the *UM10204 I²C-Bus Specification and User Manual*, Rev. 03—19 June 2007, when selecting pull-up resistor values to ensure proper operation.

Table 10. I²C Digital Input/Output Voltage

Parameter	Limit[1]	Unit
Digital Input Voltage		
Low Level Input Voltage (V_{IL})	$0.25 \times V_{DD\ I/O}$	V max
High Level Input Voltage (V_{IH})	$0.75 \times V_{DD\ I/O}$	V min
Digital Output Voltage		
Low Level Output Voltage (V_{OL})[2]	$0.2 \times V_{DD\ I/O}$	V max

[1] Limits based on characterization results; not production tested.
[2] The limit given is only for $V_{DD\ I/O} < 2$ V. When $V_{DD\ I/O} > 2$ V, the limit is 0.4 V max.

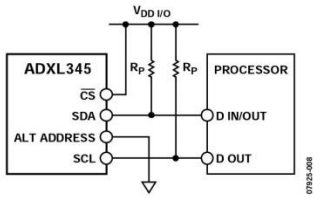


Figure 8. I²C Connection Diagram (Address 0x53)

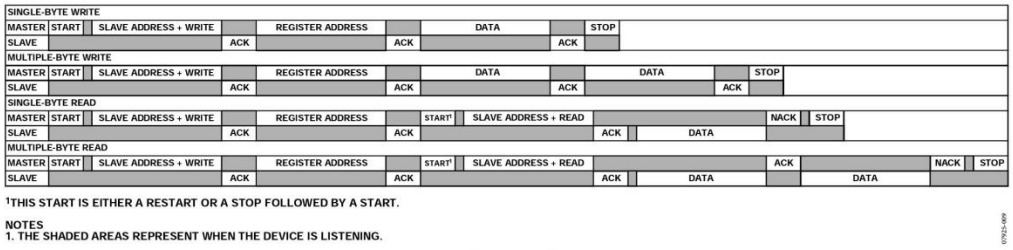


Figure 9. I²C Device Addressing

Table 11. I²C Timing (T_A = 25°C, V_S = 2.5 V, $V_{DD\ I/O}$ = 1.8 V)

	Limit[1, 2]			
Parameter	**Min**	**Max**	**Unit**	**Description**
f_{SCL}		400	kHz	SCL clock frequency
t_1	2.5		µs	SCL cycle time
t_2	0.6		µs	t_{HIGH}, SCL high time
t_3	1.3		µs	t_{LOW}, SCL low time
t_4	0.6		µs	$t_{HD,\ STA}$, start/repeated start condition hold time
t_5	350		ns	$t_{SU,\ DAT}$, data setup time
t_6[3, 4, 5, 6]	0	0.65	µs	$t_{HD,\ DAT}$, data hold time
t_7	0.6		µs	$t_{SU,\ STA}$, setup time for repeated start
t_8	0.6		µs	$t_{SU,\ STO}$, stop condition setup time
t_9	1.3		µs	t_{BUF}, bus-free time between a stop condition and a start condition
t_{10}		300	ns	t_R, rise time of both SCL and SDA when receiving
	0		ns	t_R, rise time of both SCL and SDA when receiving or transmitting
t_{11}		250	ns	t_F, fall time of SDA when receiving
		300	ns	t_F, fall time of both SCL and SDA when transmitting
	20 + 0.1 C_b[7]		ns	t_F, fall time of both SCL and SDA when transmitting or receiveing
C_b		400	pF	Capacitive load for each bus line

[1] Limits based on characterization results, with f_{SCL} = 400 kHz and a 3 mA sink current; not production tested.
[2] All values referred to the V_{IH} and the V_{IL} levels given in Table 10.
[3] t_6 is the data hold time that is measured from the falling edge of SCL. It applies to data in transmission and acknowledge times.
[4] A transmitting device must internally provide an output hold time of at least 300 ns for the SDA signal (with respect to $V_{IH(min)}$ of the SCL signal) to bridge the undefined region of the falling edge of SCL.
[5] The maximum t_6 value must be met only if the device does not stretch the low period (t_3) of the SCL signal.
[6] The maximum value for t_6 is a function of the clock low time (t_3), the clock rise time (t_{10}), and the minimum data setup time ($t_{5(min)}$). This value is calculated as $t_{6(max)} = t_3 - t_{10} - t_{5(min)}$.
[7] C_b is the total capacitance of one bus line in picofarads.

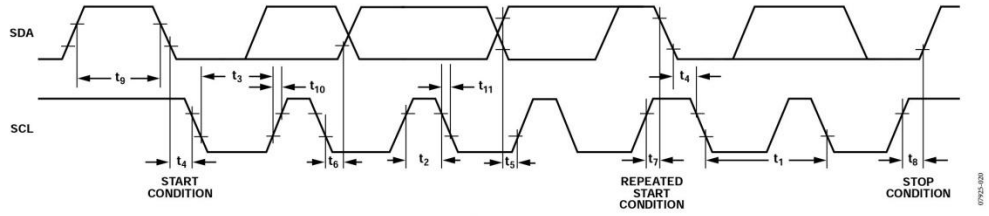


Figure 10. I²C Timing Diagram

INTERRUPTS

The ADXL345 provides two output pins for driving interrupts: INT1 and INT2. Each interrupt function is described in detail in this section. All functions can be used simultaneously, with the only limiting feature being that some functions may need to share interrupt pins. Interrupts are enabled by setting the appropriate bit in the INT_ENABLE register (Address 0x2E) and are mapped to either the INT1 or INT2 pin based on the contents of the INT_MAP register (Address 0x2F). It is recommended that interrupt bits be configured with the interrupts disabled, preventing interrupts from being accidentally triggered during configuration. This can be done by writing a value of 0x00 to the INT_ENABLE register. Clearing interrupts is performed either by reading the data registers (Address 0x32 to Address 0x37) until the interrupt condition is no longer valid for the data-related interrupts or by reading the INT_SOURCE register (Address 0x30) for the remaining interrupts. This section describes the interrupts that can be set in the INT_ENABLE register and monitored in the INT_SOURCE register.

DATA_READY

The DATA_READY bit is set when new data is available and is cleared when no new data is available.

SINGLE_TAP

The SINGLE_TAP bit is set when a single acceleration event that is greater than the value in the THRESH_TAP register (Address 0x1D) occurs for less time than is specified in the DUR register (Address 0x21).

DOUBLE_TAP

The DOUBLE_TAP bit is set when two acceleration events that are greater than the value in the THRESH_TAP register (Address 0x1D) occur for less time than is specified in the DUR register (Address 0x21), with the second tap starting after the time specified by the latent register (Address 0x22) but within the time specified in the window register (Address 0x23). See the Tap Detection section for more details.

Activity

The activity bit is set when acceleration greater than the value stored in the THRESH_ACT register (Address 0x24) is experienced.

Inactivity

The inactivity bit is set when acceleration of less than the value stored in the THRESH_INACT register (Address 0x25) is experienced for more time than is specified in the TIME_INACT register (Address 0x26). The maximum value for TIME_INACT is 255 sec.

FREE_FALL

The FREE_FALL bit is set when acceleration of less than the value stored in the THRESH_FF register (Address 0x28) is experienced for more time than is specified in the TIME_FF register (Address 0x29). The FREE_FALL interrupt differs from the inactivity interrupt as follows: all axes always participate, the timer period is much smaller (1.28 sec maximum), and the mode of operation is always dc-coupled.

Watermark

The watermark bit is set when the number of samples in FIFO equals the value stored in the samples bits (Register FIFO_CTL, Address 0x38). The watermark bit is cleared automatically when FIFO is read, and the content returns to a value below the value stored in the samples bits.

Overrun

The overrun bit is set when new data replaces unread data. The precise operation of the overrun function depends on the FIFO mode. In bypass mode, the overrun bit is set when new data replaces unread data in the DATAX, DATAY, and DATAZ registers (Address 0x32 to Address 0x37). In all other modes, the overrun bit is set when FIFO is filled. The overrun bit is automatically cleared when the contents of FIFO are read.

FIFO

The ADXL345 contains patent pending technology for an embedded 32-level FIFO that can be used to minimize host processor burden. This buffer has four modes: bypass, FIFO, stream, and trigger (see Table 19). Each mode is selected by the settings of the FIFO_MODE bits in the FIFO_CTL register (Address 0x38).

Bypass Mode

In bypass mode, FIFO is not operational and, therefore, remains empty.

FIFO Mode

In FIFO mode, data from measurements of the x-, y-, and z-axes are stored in FIFO. When the number of samples in FIFO equals the level specified in the samples bits of the FIFO_CTL register (Address 0x38), the watermark interrupt is set. FIFO continues accumulating samples until it is full (32 samples from measurements of the x-, y-, and z-axes) and then stops collecting data. After FIFO stops collecting data, the device continues to operate; therefore, features such as tap detection can be used after FIFO is full. The watermark interrupt continues to occur until the number of samples in FIFO is less than the value stored in the samples bits of the FIFO_CTL register.

Stream Mode

In stream mode, data from measurements of the x-, y-, and z-axes are stored in FIFO. When the number of samples in FIFO equals the level specified in the samples bits of the FIFO_CTL register (Address 0x38), the watermark interrupt is set. FIFO continues accumulating samples and holds the latest 32 samples from measurements of the x-, y-, and z-axes, discarding older data as new data arrives. The watermark interrupt continues occurring until the number of samples in FIFO is less than the value stored in the samples bits of the FIFO_CTL register.

Trigger Mode

In trigger mode, FIFO accumulates samples, holding the latest 32 samples from measurements of the x-, y-, and z-axes. After a trigger event occurs and an interrupt is sent to the INT1 or INT2 pin (determined by the trigger bit in the FIFO_CTL register), FIFO keeps the last n samples (where n is the value specified by the samples bits in the FIFO_CTL register) and then operates in FIFO mode, collecting new samples only when FIFO is not full. A delay of at least 5 µs should be present between the trigger event occurring and the start of reading data from the FIFO to allow the FIFO to discard and retain the necessary samples. Additional trigger events cannot be recognized until the trigger mode is reset. To reset the trigger mode, set the device to bypass mode and then set the device back to trigger mode. Note that the FIFO data should be read first because placing the device into bypass mode clears FIFO.

Retrieving Data from FIFO

The FIFO data is read through the DATAX, DATAY, and DATAZ registers (Address 0x32 to Address 0x37). When the FIFO is in FIFO, stream, or trigger mode, reads to the DATAX, DATAY, and DATAZ registers read data stored in the FIFO. Each time data is read from the FIFO, the oldest x-, y-, and z-axes data are placed into the DATAX, DATAY and DATAZ registers.

If a single-byte read operation is performed, the remaining bytes of data for the current FIFO sample are lost. Therefore, all axes of interest should be read in a burst (or multiple-byte) read operation. To ensure that the FIFO has completely popped (that is, that new data has completely moved into the DATAX, DATAY, and DATAZ registers), there must be at least 5 µs between the end of reading the data registers and the start of a new read of the FIFO or a read of the FIFO_STATUS register (Address 0x39). The end of reading a data register is signified by the transition from Register 0x37 to Register 0x38 or by the $\overline{CS}$ pin going high.

For SPI operation at 1.6 MHz or less, the register addressing portion of the transmission is a sufficient delay to ensure that the FIFO has completely popped. For SPI operation greater than 1.6 MHz, it is necessary to deassert the $\overline{CS}$ pin to ensure a total delay of 5 µs; otherwise, the delay will not be sufficient. The total delay necessary for 5 MHz operation is at most 3.4 µs. This is not a concern when using I^2C mode because the communication rate is low enough to ensure a sufficient delay between FIFO reads.

SELF-TEST

The ADXL345 incorporates a self-test feature that effectively tests its mechanical and electronic systems simultaneously. When the self-test function is enabled (via the SELF_TEST bit in the DATA_FORMAT register, Address 0x31), an electrostatic force is exerted on the mechanical sensor. This electrostatic force moves the mechanical sensing element in the same manner as acceleration, and it is additive to the acceleration experienced by the device. This added electrostatic force results in an output change in the x-, y-, and z-axes. Because the electrostatic force is proportional to V_S^2, the output change varies with V_S. The self-test feature of the ADXL345 also exhibits a bimodal behavior that depends on which phase of the clock self-test is enabled. However, the limits shown in Table 1 and Table 12 to Table 15 are valid for all potential self-test values across the entire allowable voltage range. Use of the self-test feature at data rates less than 100 Hz may yield values outside these limits. Therefore, the part should be placed into a data rate of 100 Hz or greater when using self-test.

Table 12. Self-Test Output in LSB for ±2 *g*, Full Resolution

Axis	Min	Max	Unit
X	50	540	LSB
Y	−540	−50	LSB
Z	75	875	LSB

Table 13. Self-Test Output in LSB for ±4 *g*, 10-Bit Resolution

Axis	Min	Max	Unit
X	25	270	LSB
Y	−270	−25	LSB
Z	38	438	LSB

Table 14. Self-Test Output in LSB for ±8 *g*, 10-Bit Resolution

Axis	Min	Max	Unit
X	12	135	LSB
Y	−135	−12	LSB
Z	19	219	LSB

Table 15. Self-Test Output in LSB for ±16 *g*, 10-Bit Resolution

Axis	Min	Max	Unit
X	6	67	LSB
Y	−67	−6	LSB
Z	10	110	LSB

REGISTER MAP

Table 16. Register Map

Address					
Hex	**Dec**	**Name**	**Type**	**Reset Value**	**Description**
0x00	0	DEVID	R	11100101	Device ID.
0x01 to 0x01C	1 to 28	Reserved			Reserved. Do not access.
0x1D	29	THRESH_TAP	R/$\overline{W}$	00000000	Tap threshold.
0x1E	30	OFSX	R/$\overline{W}$	00000000	X-axis offset.
0x1F	31	OFSY	R/$\overline{W}$	00000000	Y-axis offset.
0x20	32	OFSZ	R/$\overline{W}$	00000000	Z-axis offset.
0x21	33	DUR	R/$\overline{W}$	00000000	Tap duration.
0x22	34	Latent	R/$\overline{W}$	00000000	Tap latency.
0x23	35	Window	R/$\overline{W}$	00000000	Tap window.
0x24	36	THRESH_ACT	R/$\overline{W}$	00000000	Activity threshold.
0x25	37	THRESH_INACT	R/$\overline{W}$	00000000	Inactivity threshold.
0x26	38	TIME_INACT	R/$\overline{W}$	00000000	Inactivity time.
0x27	39	ACT_INACT_CTL	R/$\overline{W}$	00000000	Axis enable control for activity and inactivity detection.
0x28	40	THRESH_FF	R/$\overline{W}$	00000000	Free-fall threshold.
0x29	41	TIME_FF	R/$\overline{W}$	00000000	Free-fall time.
0x2A	42	TAP_AXES	R/$\overline{W}$	00000000	Axis control for tap/double tap.
0x2B	43	ACT_TAP_STATUS	R	00000000	Source of tap/double tap.
0x2C	44	BW_RATE	R/$\overline{W}$	00001010	Data rate and power mode control.
0x2D	45	POWER_CTL	R/$\overline{W}$	00000000	Power-saving features control.
0x2E	46	INT_ENABLE	R/$\overline{W}$	00000000	Interrupt enable control.
0x2F	47	INT_MAP	R/$\overline{W}$	00000000	Interrupt mapping control.
0x30	48	INT_SOURCE	R	00000010	Source of interrupts.
0x31	49	DATA_FORMAT	R/$\overline{W}$	00000000	Data format control.
0x32	50	DATAX0	R	00000000	X-Axis Data 0.
0x33	51	DATAX1	R	00000000	X-Axis Data 1.
0x34	52	DATAY0	R	00000000	Y-Axis Data 0.
0x35	53	DATAY1	R	00000000	Y-Axis Data 1.
0x36	54	DATAZ0	R	00000000	Z-Axis Data 0.
0x37	55	DATAZ1	R	00000000	Z-Axis Data 1.
0x38	56	FIFO_CTL	R/$\overline{W}$	00000000	FIFO control.
0x39	57	FIFO_STATUS	R	00000000	FIFO status.

REGISTER DEFINITIONS

Register 0x00—DEVID (Read Only)

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	0	0	1	0	1

The DEVID register holds a fixed device ID code of 0xE5 (345 octal).

Register 0x1D—THRESH_TAP (Read/Write)

The THRESH_TAP register is eight bits and holds the threshold value for tap interrupts. The data format is unsigned, so the magnitude of the tap event is compared with the value in THRESH_TAP. The scale factor is 62.5 mg/LSB (that is, 0xFF = +16 *g*). A value of 0 may result in undesirable behavior if tap/double tap interrupts are enabled.

Register 0x1E, Register 0x1F, Register 0x20—OFSX, OFSY, OFSZ (Read/Write)

The OFSX, OFSY, and OFSZ registers are each eight bits and offer user-set offset adjustments in twos complement format with a scale factor of 15.6 mg/LSB (that is, 0x7F = +2 *g*).

Register 0x21—DUR (Read/Write)

The DUR register is eight bits and contains an unsigned time value representing the maximum time that an event must be above the THRESH_TAP threshold to qualify as a tap event. The scale factor is 625 µs/LSB. A value of 0 disables the tap/double tap functions.

Register 0x22—Latent (Read/Write)

The latent register is eight bits and contains an unsigned time value representing the wait time from the detection of a tap event to the start of the time window (defined by the window register) during which a possible second tap event can be detected. The scale factor is 1.25 ms/LSB. A value of 0 disables the double tap function.

Register 0x23—Window (Read/Write)

The window register is eight bits and contains an unsigned time value representing the amount of time after the expiration of the latency time (determined by the latent register) during which a second valid tap can begin. The scale factor is 1.25 ms/LSB. A value of 0 disables the double tap function.

Register 0x24—THRESH_ACT (Read/Write)

The THRESH_ACT register is eight bits and holds the threshold value for detecting activity. The data format is unsigned, so the magnitude of the activity event is compared with the value in the THRESH_ACT register. The scale factor is 62.5 mg/LSB. A value of 0 may result in undesirable behavior if the activity interrupt is enabled.

Register 0x25—THRESH_INACT (Read/Write)

The THRESH_INACT register is eight bits and holds the threshold value for detecting inactivity. The data format is unsigned, so the magnitude of the inactivity event is compared with the value in the THRESH_INACT register. The scale factor is 62.5 mg/LSB. A value of 0 mg may result in undesirable behavior if the inactivity interrupt is enabled.

Register 0x26—TIME_INACT (Read/Write)

The TIME_INACT register is eight bits and contains an unsigned time value representing the amount of time that acceleration must be less than the value in the THRESH_INACT register for inactivity to be declared. The scale factor is 1 sec/LSB. Unlike the other interrupt functions, which use unfiltered data (see the Threshold section), the inactivity function uses filtered output data. At least one output sample must be generated for the inactivity interrupt to be triggered. This results in the function appearing unresponsive if the TIME_INACT register is set to a value less than the time constant of the output data rate. A value of 0 results in an interrupt when the output data is less than the value in the THRESH_INACT register.

Register 0x27—ACT_INACT_CTL (Read/Write)

D7	D6	D5	D4
ACT ac/dc	ACT_X enable	ACT_Y enable	ACT_Z enable
D3	**D2**	**D1**	**D0**
INACT ac/dc	INACT_X enable	INACT_Y enable	INACT_Z enable

ACT AC/DC and INACT AC/DC Bits

A setting of 0 selects dc-coupled operation, and a setting of 1 enables ac-coupled operation. In dc-coupled operation, the current acceleration magnitude is compared directly with THRESH_ACT and THRESH_INACT to determine whether activity or inactivity is detected.

In ac-coupled operation for activity detection, the acceleration value at the start of activity detection is taken as a reference value. New samples of acceleration are then compared to this reference value, and if the magnitude of the difference exceeds the THRESH_ACT value, the device triggers an activity interrupt.

Similarly, in ac-coupled operation for inactivity detection, a reference value is used for comparison and is updated whenever the device exceeds the inactivity threshold. After the reference value is selected, the device compares the magnitude of the difference between the reference value and the current acceleration with THRESH_INACT. If the difference is less than the value in THRESH_INACT for the time in TIME_INACT, the device is considered inactive and the inactivity interrupt is triggered.

ACT_x Enable Bits and INACT_x Enable Bits

A setting of 1 enables x-, y-, or z-axis participation in detecting activity or inactivity. A setting of 0 excludes the selected axis from participation. If all axes are excluded, the function is disabled.

Register 0x28—THRESH_FF (Read/Write)

The THRESH_FF register is eight bits and holds the threshold value, in unsigned format, for free-fall detection. The root-sum-square (RSS) value of all axes is calculated and compared with the value in THRESH_FF to determine if a free-fall event occurred. The scale factor is 62.5 mg/LSB. Note that a value of 0 mg may result in undesirable behavior if the free-fall interrupt is enabled. Values between 300 mg and 600 mg (0x05 to 0x09) are recommended.

Register 0x29—TIME_FF (Read/Write)

The TIME_FF register is eight bits and stores an unsigned time value representing the minimum time that the RSS value of all axes must be less than THRESH_FF to generate a free-fall interrupt. The scale factor is 5 ms/LSB. A value of 0 may result in undesirable behavior if the free-fall interrupt is enabled. Values between 100 ms and 350 ms (0x14 to 0x46) are recommended.

Register 0x2A—TAP_AXES (Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	Suppress	TAP_X enable	TAP_Y enable	TAP_Z enable

Suppress Bit

Setting the suppress bit suppresses double tap detection if acceleration greater than the value in THRESH_TAP is present between taps. See the Tap Detection section for more details.

TAP_x Enable Bits

A setting of 1 in the TAP_X enable, TAP_Y enable, or TAP_Z enable bit enables x-, y-, or z-axis participation in tap detection. A setting of 0 excludes the selected axis from participation in tap detection.

Register 0x2B—ACT_TAP_STATUS (Read Only)

D7	D6	D5	D4	D3	D2	D1	D0
0	ACT_X source	ACT_Y source	ACT_Z source	Asleep	TAP_X source	TAP_Y source	TAP_Z source

ACT_x Source and TAP_x Source Bits

These bits indicate the first axis involved in a tap or activity event. A setting of 1 corresponds to involvement in the event, and a setting of 0 corresponds to no involvement. When new data is available, these bits are not cleared but are overwritten by the new data. The ACT_TAP_STATUS register should be read before clearing the interrupt. Disabling an axis from participation clears the corresponding source bit when the next activity or tap/double tap event occurs.

Asleep Bit

A setting of 1 in the asleep bit indicates that the part is asleep, and a setting of 0 indicates that the part is not asleep. See the Register 0x2D—POWER_CTL (Read/Write) section for more information on autosleep mode.

Register 0x2C—BW_RATE (Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	LOW_POWER	Rate			

LOW_POWER Bit

A setting of 0 in the LOW_POWER bit selects normal operation, and a setting of 1 selects reduced power operation, which has somewhat higher noise (see the Power Modes section for details).

Rate Bits

These bits select the device bandwidth and output data rate (see Table 6 and Table 7 for details). The default value is 0x0A, which translates to a 100 Hz output data rate. An output data rate should be selected that is appropriate for the communication protocol and frequency selected. Selecting too high of an output data rate with a low communication speed results in samples being discarded.

Register 0x2D—POWER_CTL (Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0
0	0	Link	AUTO_SLEEP	Measure	Sleep	Wakeup	

Link Bit

A setting of 1 in the link bit with both the activity and inactivity functions enabled delays the start of the activity function until inactivity is detected. After activity is detected, inactivity detection begins, preventing the detection of activity. This bit serially links the activity and inactivity functions. When this bit is set to 0, the inactivity and activity functions are concurrent. Additional information can be found in the Link Mode section.

When clearing the link bit, it is recommended that the part be placed into standby mode and then set back to measurement mode with a subsequent write. This is done to ensure that the device is properly biased if sleep mode is manually disabled; otherwise, the first few samples of data after the link bit is cleared may have additional noise, especially if the device was asleep when the bit was cleared.

AUTO_SLEEP Bit

If the link bit is set, a setting of 1 in the AUTO_SLEEP bit sets the ADXL345 to switch to sleep mode when inactivity is detected (that is, when acceleration has been below the THRESH_INACT value for at least the time indicated by TIME_INACT). A setting of 0 disables automatic switching to sleep mode. See the description of the sleep bit in this section for more information.

When clearing the AUTO_SLEEP bit, it is recommended that the part be placed into standby mode and then set back to measurement mode with a subsequent write. This is done to ensure that the device is properly biased if sleep mode is manually disabled; otherwise, the first few samples of data after the AUTO_SLEEP bit is cleared may have additional noise, especially if the device was asleep when the bit was cleared.

Measure Bit

A setting of 0 in the measure bit places the part into standby mode, and a setting of 1 places the part into measurement mode. The ADXL345 powers up in standby mode with minimum power consumption.

Sleep Bit

A setting of 0 in the sleep bit puts the part into the normal mode of operation, and a setting of 1 places the part into sleep mode. Sleep mode suppresses DATA_READY, stops transmission of data to FIFO, and switches the sampling rate to one specified by the wakeup bits. In sleep mode, only the activity function can be used.

When clearing the sleep bit, it is recommended that the part be placed into standby mode and then set back to measurement mode with a subsequent write. This is done to ensure that the device is properly biased if sleep mode is manually disabled; otherwise, the first few samples of data after the sleep bit is cleared may have additional noise, especially if the device was asleep when the bit was cleared.

Wakeup Bits

These bits control the frequency of readings in sleep mode as described in Table 17.

Table 17. Frequency of Readings in Sleep Mode

Setting		
D1	**D0**	**Frequency (Hz)**
0	0	8
0	1	4
1	0	2
1	1	1

Register 0x2E—INT_ENABLE (Read/Write)

D7	**D6**	**D5**	**D4**
DATA_READY	SINGLE_TAP	DOUBLE_TAP	Activity
D3	**D2**	**D1**	**D0**
Inactivity	FREE_FALL	Watermark	Overrun

Setting bits in this register to a value of 1 enables their respective functions to generate interrupts, whereas a value of 0 prevents the functions from generating interrupts. The DATA_READY, watermark, and overrun bits enable only the interrupt output; the functions are always enabled. It is recommended that interrupts be configured before enabling their outputs.

Register 0x2F—INT_MAP (Read/Write)

D7	**D6**	**D5**	**D4**
DATA_READY	SINGLE_TAP	DOUBLE_TAP	Activity
D3	**D2**	**D1**	**D0**
Inactivity	FREE_FALL	Watermark	Overrun

Any bits set to 0 in this register send their respective interrupts to the INT1 pin, whereas bits set to 1 send their respective interrupts to the INT2 pin. All selected interrupts for a given pin are ORed.

Register 0x30—INT_SOURCE (Read Only)

D7	**D6**	**D5**	**D4**
DATA_READY	SINGLE_TAP	DOUBLE_TAP	Activity
D3	**D2**	**D1**	**D0**
Inactivity	FREE_FALL	Watermark	Overrun

Bits set to 1 in this register indicate that their respective functions have triggered an event, whereas a value of 0 indicates that the corresponding event has not occurred. The DATA_READY, watermark, and overrun bits are always set if the corresponding events occur, regardless of the INT_ENABLE register settings, and are cleared by reading data from the DATAX, DATAY, and DATAZ registers. The DATA_READY and watermark bits may require multiple reads, as indicated in the FIFO mode descriptions in the FIFO section. Other bits, and the corresponding interrupts, are cleared by reading the INT_SOURCE register.

Register 0x31—DATA_FORMAT (Read/Write)

D7	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**
SELF_TEST	SPI	INT_INVERT	0	FULL_RES	Justify	Range	

The DATA_FORMAT register controls the presentation of data to Register 0x32 through Register 0x37. All data, except that for the ±16 *g* range, must be clipped to avoid rollover.

SELF_TEST Bit

A setting of 1 in the SELF_TEST bit applies a self-test force to the sensor, causing a shift in the output data. A value of 0 disables the self-test force.

SPI Bit

A value of 1 in the SPI bit sets the device to 3-wire SPI mode, and a value of 0 sets the device to 4-wire SPI mode.

INT_INVERT Bit

A value of 0 in the INT_INVERT bit sets the interrupts to active high, and a value of 1 sets the interrupts to active low.

FULL_RES Bit

When this bit is set to a value of 1, the device is in full resolution mode, where the output resolution increases with the *g* range set by the range bits to maintain a 4 mg/LSB scale factor. When the FULL_RES bit is set to 0, the device is in 10-bit mode, and the range bits determine the maximum *g* range and scale factor.

Justify Bit

A setting of 1 in the justify bit selects left (MSB) justified mode, and a setting of 0 selects right justified mode with sign extension.

Range Bits

These bits set the *g* range as described in Table 18.

Table 18. *g* Range Setting

Setting		
D1	**D0**	***g* Range**
0	0	±2 *g*
0	1	±4 *g*
1	0	±8 *g*
1	1	±16 *g*

Register 0x32 to Register 0x37—DATAX0, DATAX1, DATAY0, DATAY1, DATAZ0, DATAZ1 (Read Only)

These six bytes (Register 0x32 to Register 0x37) are eight bits each and hold the output data for each axis. Register 0x32 and Register 0x33 hold the output data for the x-axis, Register 0x34 and Register 0x35 hold the output data for the y-axis, and Register 0x36 and Register 0x37 hold the output data for the z-axis. The output data is twos complement, with DATAx0 as the least significant byte and DATAx1 as the most significant byte, where x represent X, Y, or Z. The DATA_FORMAT register (Address 0x31) controls the format of the data. It is recommended that a multiple-byte read of all registers be performed to prevent a change in data between reads of sequential registers.

Register 0x38—FIFO_CTL (Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0
FIFO_MODE		Trigger	Samples				

FIFO_MODE Bits

These bits set the FIFO mode, as described in Table 19.

Table 19. FIFO Modes

Setting			
D7	D6	Mode	Function
0	0	Bypass	FIFO is bypassed.
0	1	FIFO	FIFO collects up to 32 values and then stops collecting data, collecting new data only when FIFO is not full.
1	0	Stream	FIFO holds the last 32 data values. When FIFO is full, the oldest data is overwritten with newer data.
1	1	Trigger	When triggered by the trigger bit, FIFO holds the last data samples before the trigger event and then continues to collect data until full. New data is collected only when FIFO is not full.

Trigger Bit

A value of 0 in the trigger bit links the trigger event of trigger mode to INT1, and a value of 1 links the trigger event to INT2.

Samples Bits

The function of these bits depends on the FIFO mode selected (see Table 20). Entering a value of 0 in the samples bits immediately sets the watermark status bit in the INT_SOURCE register, regardless of which FIFO mode is selected. Undesirable operation may occur if a value of 0 is used for the samples bits when trigger mode is used.

Table 20. Samples Bits Functions

FIFO Mode	Samples Bits Function
Bypass	None.
FIFO	Specifies how many FIFO entries are needed to trigger a watermark interrupt.
Stream	Specifies how many FIFO entries are needed to trigger a watermark interrupt.
Trigger	Specifies how many FIFO samples are retained in the FIFO buffer before a trigger event.

0x39—FIFO_STATUS (Read Only)

D7	D6	D5	D4	D3	D2	D1	D0
FIFO_TRIG	0	Entries					

FIFO_TRIG Bit

A 1 in the FIFO_TRIG bit corresponds to a trigger event occurring, and a 0 means that a FIFO trigger event has not occurred.

Entries Bits

These bits report how many data values are stored in FIFO. Access to collect the data from FIFO is provided through the DATAX, DATAY, and DATAZ registers. FIFO reads must be done in burst or multiple-byte mode because each FIFO level is cleared after any read (single- or multiple-byte) of FIFO. FIFO stores a maximum of 32 entries, which equates to a maximum of 33 entries available at any given time because an additional entry is available at the output filter of the device.

APPLICATIONS INFORMATION

POWER SUPPLY DECOUPLING

A 1 µF tantalum capacitor (C_S) at V_S and a 0.1 µF ceramic capacitor (C_{IO}) at $V_{DD\ I/O}$ placed close to the ADXL345 supply pins is used for testing and is recommended to adequately decouple the accelerometer from noise on the power supply. If additional decoupling is necessary, a resistor or ferrite bead, no larger than 100 Ω, in series with V_S may be helpful. Additionally, increasing the bypass capacitance on V_S to a 10 µF tantalum capacitor in parallel with a 0.1 µF ceramic capacitor may also improve noise.

Care should be taken to ensure that the connection from the ADXL345 ground to the power supply ground has low impedance because noise transmitted through ground has an effect similar to noise transmitted through V_S. It is recommended that V_S and $V_{DD\ I/O}$ be separate supplies to minimize digital clocking noise on the V_S supply. If this is not possible, additional filtering of the supplies as previously mentioned may be necessary.

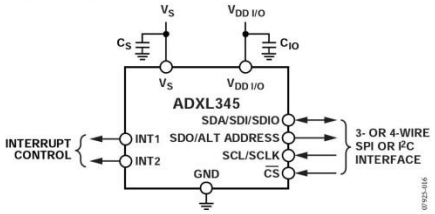


Figure 11. Application Diagram

MECHANICAL CONSIDERATIONS FOR MOUNTING

The ADXL345 should be mounted on the PCB in a location close to a hard mounting point of the PCB to the case. Mounting the ADXL345 at an unsupported PCB location, as shown in Figure 12, may result in large, apparent measurement errors due to undampened PCB vibration. Locating the accelerometer near a hard mounting point ensures that any PCB vibration at the accelerometer is above the accelerometer's mechanical sensor resonant frequency and, therefore, effectively invisible to the accelerometer.

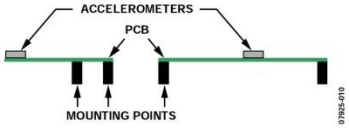


Figure 12. Incorrectly Placed Accelerometers

TAP DETECTION

The tap interrupt function is capable of detecting either single or double taps. The following parameters are shown in Figure 13 for a valid single and valid double tap event:

- The tap detection threshold is defined by the THRESH_TAP register (Address 0x1D).
- The maximum tap duration time is defined by the DUR register (Address 0x21).
- The tap latency time is defined by the latent register (Address 0x22) and is the waiting period from the end of the first tap until the start of the time window, when a second tap can be detected, which is determined by the value in the window register (Address 0x23).
- The interval after the latency time (set by the latent register) is defined by the window register. Although a second tap must begin after the latency time has expired, it need not finish before the end of the time defined by the window register.

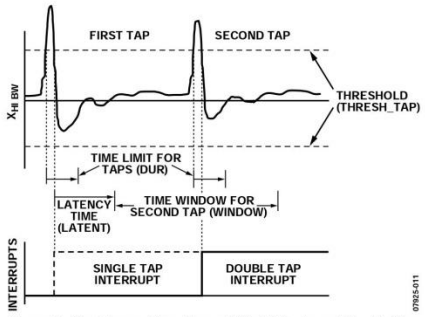


Figure 13. Tap Interrupt Function with Valid Single and Double Taps

If only the single tap function is in use, the single tap interrupt is triggered when the acceleration goes below the threshold, as long as DUR has not been exceeded. If both single and double tap functions are in use, the single tap interrupt is triggered when the double tap event has been either validated or invalidated.

Several events can occur to invalidate the second tap of a double tap event. First, if the suppress bit in the TAP_AXES register (Address 0x2A) is set, any acceleration spike above the threshold during the latency time (set by the latent register) invalidates the double tap detection, as shown in Figure 14.

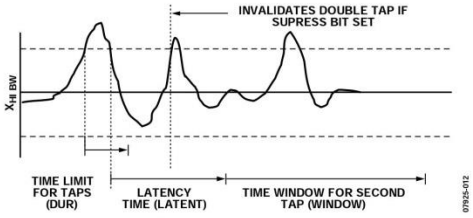


Figure 14. Double Tap Event Invalid Due to High g Event When the Suppress Bit Is Set

A double tap event can also be invalidated if acceleration above the threshold is detected at the start of the time window for the second tap (set by the window register). This results in an invalid double tap at the start of this window, as shown in Figure 15. Additionally, a double tap event can be invalidated if an accel-

eration exceeds the time limit for taps (set by the DUR register), resulting in an invalid double tap at the end of the DUR time limit for the second tap event, also shown in Figure 15.

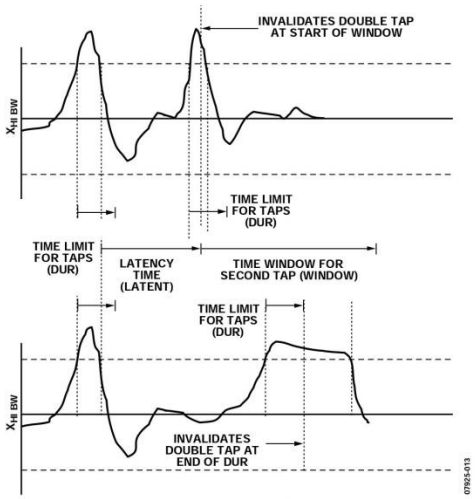


Figure 15. Tap Interrupt Function with Invalid Double Taps

Single taps, double taps, or both can be detected by setting the respective bits in the INT_ENABLE register (Address 0x2E). Control over participation of each of the three axes in single tap/double tap detection is exerted by setting the appropriate bits in the TAP_AXES register (Address 0x2A). For the double tap function to operate, both the latent and window registers must be set to a nonzero value.

Every mechanical system has somewhat different single tap/double tap responses based on the mechanical characteristics of the system. Therefore, some experimentation with values for the latent, window, and THRESH_TAP registers is required. In general, a good starting point is to set the latent register to a value greater than 0x10, to set the window register to a value greater than 0x10, and to set the THRESH_TAP register to be greater than 3 *g*. Setting a very low value in the latent, window, or THRESH_TAP register may result in an unpredictable response due to the accelerometer picking up echoes of the tap inputs.

After a tap interrupt has been received, the first axis to exceed the THRESH_TAP level is reported in the ACT_TAP_STATUS register (Address 0x2B). This register is never cleared, but is overwritten with new data.

THRESHOLD

The lower output data rates are achieved by decimating a common sampling frequency inside the device. The activity, free-fall, and single tap/double tap detection functions are performed using unfiltered data. Since the output data is filtered, the high frequency and high *g* data that is used to determine activity, free-fall, and single tap/double tap events may not be present if the output of the accelerometer is examined. This may result in trigger events being detected when acceleration does not appear to trigger an event because the unfiltered data may have exceeded a threshold or remained below a threshold for a certain period of time while the filtered output data has not exceeded such a threshold.

LINK MODE

The function of the link bit is to reduce the number of activity interrupts that the processor must service by setting the device to look for activity only after inactivity. For proper operation of this feature, the processor must still respond to the activity and inactivity interrupts by reading the INT_SOURCE register (Address 0x30) and, therefore, clearing the interrupts. If an activity interrupt is not cleared, the part cannot go into autosleep mode. The asleep bit in the ACT_TAP_STATUS register (Address 0x2B) indicates if the part is asleep.

SLEEP MODE VS. LOW POWER MODE

In applications where a low data rate is sufficient and low power consumption is desired, it is recommended that the low power mode be used in conjunction with the FIFO. The sleep mode, while offering a low data rate and low average current consumption, suppresses the DATA_READY interrupt, preventing the accelerometer from sending an interrupt signal to the host processor when data is ready to be collected. In this application, setting the part into low power mode (by setting the LOW_POWER bit in the BW_RATE register) and enabling the FIFO in FIFO mode to collect a large value of samples reduces the power consumption of the ADXL345 and allows the host processor to go to sleep while the FIFO is filling up.

USING SELF-TEST

The self-test change is defined as the difference between the acceleration output of an axis with self-test enabled and the acceleration output of the same axis with self-test disabled (see Endnote 4 of Table 1). This definition assumes that the sensor does not move between these two measurements, because if the sensor moves, a non–self-test related shift corrupts the test.

Proper configuration of the ADXL345 is also necessary for an accurate self-test measurement. The part should be set with a data rate greater than or equal to 100 Hz. This is done by ensuring that a value greater than or equal to 0x0A is written into the rate bits (Bit D3 through Bit D0) in the BW_RATE register (Address 0x2C). It is also recommended that the part be set to full-resolution, 16 *g* mode to ensure that there is sufficient dynamic range for the entire self-test shift. This is done by setting Bit D3 of the DATA_FORMAT register (Address 0x31) and writing a value of 0x03 to the range bits (Bit D1 and Bit D0) of the DATA_FORMAT register (Address 0x31). This results in a high dynamic range for measurement and a 3.9 mg/LSB scale factor.

After the part is configured for accurate self-test measurement, several samples of x-, y-, and z-axis acceleration data should be retrieved from the sensor and averaged together. The number of

samples averaged is a choice of the system designer, but a recommended starting point is 0.1 sec worth of data, which corresponds to 10 samples at 100 Hz data rate. The averaged values should be stored and labeled appropriately as the self-test disabled data, that is, X_{ST_OFF}, Y_{ST_OFF}, and Z_{ST_OFF}.

Next, self-test should be enabled by setting Bit D7 of the DATA_FORMAT register (Address 0x31). The output needs some time (about four samples) to settle after enabling self-test. After allowing the output to settle, several samples of the x-, y-, and z-axis acceleration data should be taken again and averaged. It is recommended that the same number of samples be taken for this average as was previously taken. These averaged values should again be stored and labeled appropriately as the value with self-test enabled, that is, X_{ST_ON}, Y_{ST_ON}, and Z_{ST_ON}. Self-test can then be disabled by clearing Bit D7 of the DATA_FORMAT register (Address 0x31).

With the stored values for self-test enabled and disabled, the self-test change is as follows:

$$X_{ST} = X_{ST\_ON} - X_{ST\_OFF}$$

$$Y_{ST} = Y_{ST\_ON} - Y_{ST\_OFF}$$

$$Z_{ST} = Z_{ST\_ON} - Z_{ST\_OFF}$$

Because the measured output for each axis is expressed in LSBs, X_{ST}, Y_{ST}, and Z_{ST} are also expressed in LSBs. These values can be converted to *g*'s of acceleration by multiplying each value by the 3.9 mg/LSB scale factor, if configured for full-resolution, 16 *g* mode. Additionally, Table 12 through Table 15 correspond to the self-test range converted to LSBs and can be compared with the measured self-test change. If the part was placed into full-resolution, 16 *g* mode, the values listed in Table 12 should be used. Although the fixed 10-bit mode or a range other than 16 *g* can be used, a different set of values, as indicated in Table 13 through Table 15, would need to be used. Using a range below 8 *g* may result in insufficient dynamic range and should be considered when selecting the range of operation for measuring self-test. In addition, note that the range in Table 1 and the values in Table 12 through Table 15 take into account all possible supply voltages, V_S, and no additional conversion due to V_S is necessary.

If the self-test change is within the valid range, the test is considered successful. Generally, a part is considered to pass if the minimum magnitude of change is achieved. However, a part that changes by more than the maximum magnitude is not necessarily a failure.

AXES OF ACCELERATION SENSITIVITY

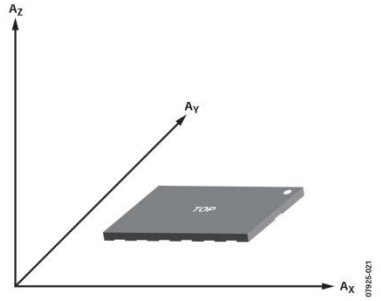


Figure 16. Axes of Acceleration Sensitivity (Corresponding Output Voltage Increases When Accelerated Along the Sensitive Axis)

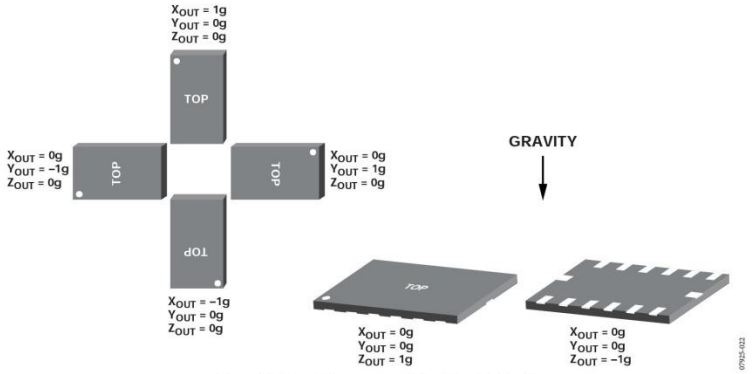


Figure 17. Output Response vs. Orientation to Gravity

LAYOUT AND DESIGN RECOMMENDATIONS

Figure 18 shows the recommended printed wiring board land pattern. Figure 19 and Table 21 provide details about the recommended soldering profile.

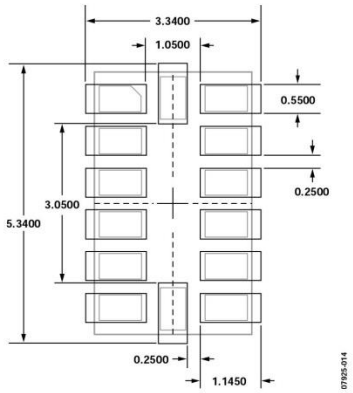


Figure 18. Recommended Printed Wiring Board Land Pattern (Dimensions shown in millimeters)

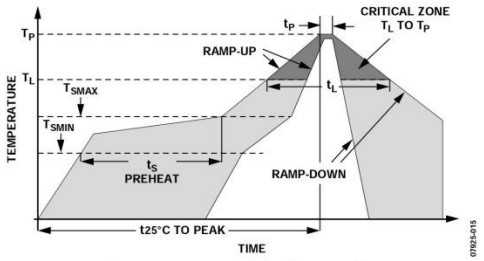


Figure 19. Recommended Soldering Profile

Table 21. Recommended Soldering Profile[1, 2]

	Condition	
Profile Feature	**Sn63/Pb37**	**Pb-Free**
Average Ramp Rate from Liquid Temperature (T_L) to Peak Temperature (T_P)	3°C/sec max	3°C/sec max
Preheat		
Minimum Temperature (T_{SMIN})	100°C	150°C
Maximum Temperature (T_{SMAX})	150°C	200°C
Time from T_{SMIN} to T_{SMAX} (t_S)	60 sec to 120 sec	60 sec to 180 sec
T_{SMAX} to T_L Ramp-Up Rate	3°C/sec max	3°C/sec max
Liquid Temperature (T_L)	183°C	217°C
Time Maintained Above T_L (t_L)	60 sec to 150 sec	60 sec to 150 sec
Peak Temperature (T_P)	240 + 0/−5°C	260 + 0/−5°C
Time of Actual T_P − 5°C (t_P)	10 sec to 30 sec	20 sec to 40 sec
Ramp-Down Rate	6°C/sec max	6°C/sec max
Time 25°C to Peak Temperature	6 minutes max	8 minutes max

[1] Based on JEDEC Standard J-STD-020D.1.
[2] For best results, the soldering profile should be in accordance with the recommendations of the manufacturer of the solder paste used.

OUTLINE DIMENSIONS

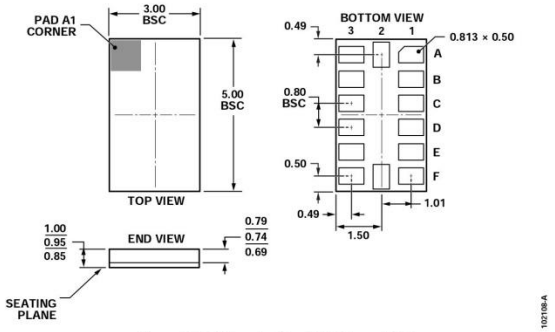


Figure 20. 14-Terminal Land Grid Array [LGA]
(CC-14-1)
Solder Terminations Finish Is Au over Ni
(Dimensions shown in millimeters)

ORDERING GUIDE

Model	Measurement Range (*g*)	Specified Voltage (V)	Temperature Range	Package Description	Package Option
ADXL345BCCZ[1]	±2, ±4, ±8, ±16	2.5	−40°C to +85°C	14-Terminal Land Grid Array [LGA]	CC-14-1
ADXL345BCCZ-RL[1]	±2, ±4, ±8, ±16	2.5	−40°C to +85°C	14-Terminal Land Grid Array [LGA]	CC-14-1
ADXL345BCCZ-RL7[1]	±2, ±4, ±8, ±16	2.5	−40°C to +85°C	14- Terminal Land Grid Array [LGA]	CC-14-1
EVAL-ADXL345Z[1]				Evaluation Board	
EVAL-ADXL345Z-M[1]				Analog Devices Inertial Sensor Evaluation System, Includes ADXL345 Satellite	
EVAL-ADXL345Z-S[1]				ADXL345 Satellite, Standalone	

[1] Z = RoHS Compliant Part.

Analog Devices offers specific products designated for automotive applications; please consult your local Analog Devices sales representative for details. Standard products sold by Analog Devices are not designed, intended, or approved for use in life support, implantable medical devices, transportation, nuclear, safety, or other equipment where malfunction of the product can reasonably be expected to result in personal injury, death, severe property damage, or severe environmental harm. Buyer uses or sells standard products for use in the above critical applications at Buyer's own risk and Buyer agrees to defend, indemnify, and hold harmless Analog Devices from any and all damages, claims, suits, or expenses resulting from such unintended use.

©2009 Analog Devices, Inc. All rights reserved. Trademarks and registered trademarks are the property of their respective owners.
D07925-0-5/09(0)

ANALOG DEVICES

www.analog.com

Anexo B. Datasheet sensor ultrasónico HC-SR04.

Tech Support: services@elecfreaks.com

Ultrasonic Ranging Module HC - SR04

Product features:

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:
(1) Using IO trigger for at least 10us high level signal,
(2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
(3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning.
Test distance = (high level time×velocity of sound (340M/S) / 2,

Wire connecting direct as following:

- 5V Supply
- Trigger Pulse Input
- Echo Pulse Output
- 0V Ground

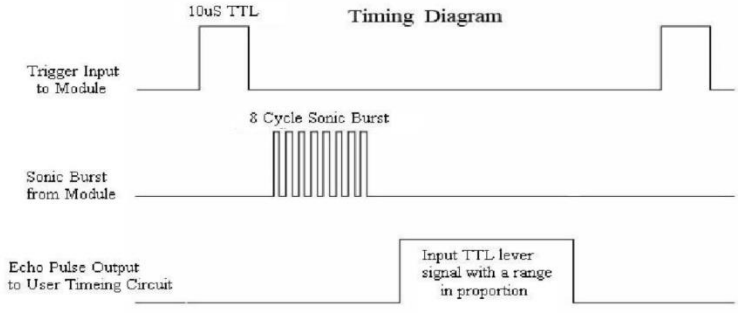
Electric Parameter

Working Voltage	**DC 5 V**
Working Current	**15mA**
Working Frequency	**40Hz**
Max Range	**4m**
Min Range	**2cm**
MeasuringAngle	**15 degree**
Trigger Input Signal	**10uS TTL pulse**
Echo Output Signal	**Input TTL lever signal and the range in proportion**
Dimension	**45*20*15mm**

Vcc Trig Echo GND

Timing diagram

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: uS / 58 = centimeters or uS / 148 =inch; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.

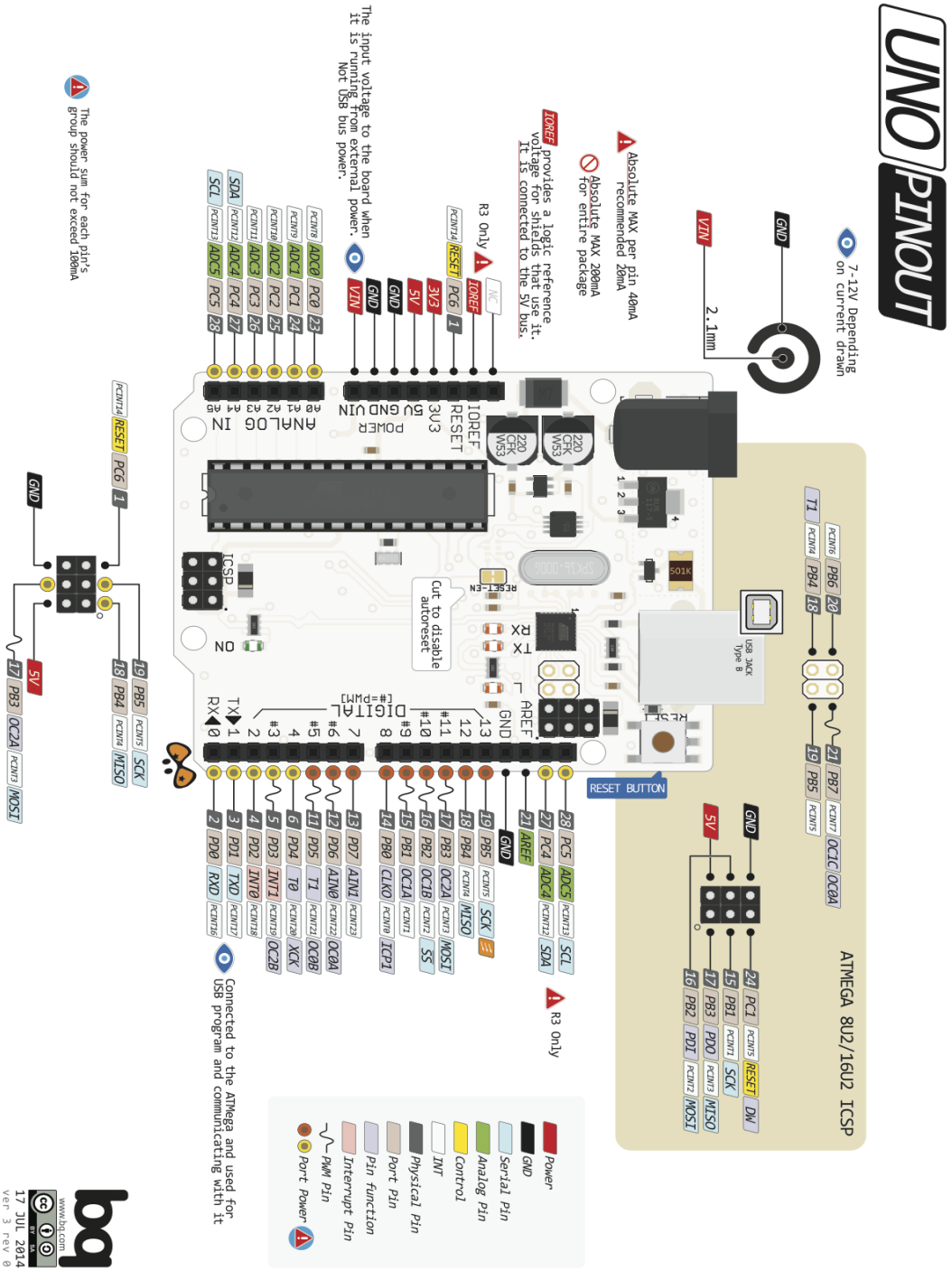


Attention:

- The module is not suggested to connect directly to electric, if connected electric, the GND terminal should be connected the module first, otherwise, it will affect the normal work of the module.
- When tested objects, the range of area is not less than 0.5 square meters and the plane requests as smooth as possible, otherwise ,it will affect the results of measuring.

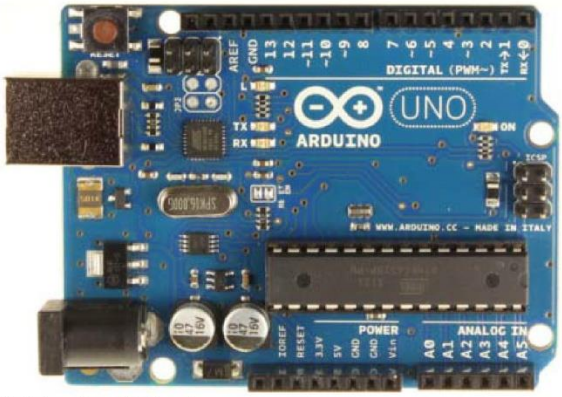
www.Elecfreaks.com

Anexo C. Arduino UNO R3 pinout Diagram

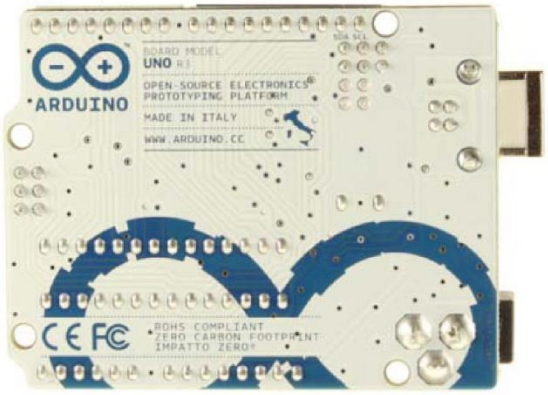


Anexo D. Datasheet Arduino UNO R3

Arduino Uno

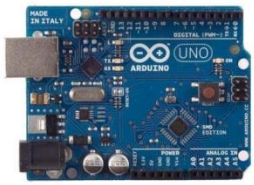


Arduino Uno R3 Front



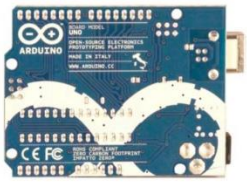
Arduino Uno R3 Back

Arduino Uno R2 Front



Arduino Uno SMD

Arduino Uno Front



Arduino Uno Back

Buy From Arduino Store Buy From Distributors

Overview

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.
Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode.
Revision 3 of the board has the following new features:

- 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes.
- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

Schematic & Reference Design

EAGLE files: arduino-uno-Rev3-reference-design.zip

Schematic: arduino-uno-Rev3-schematic.pdf

Note: The Arduino reference design can use an Atmega8, 168, or 328, Current models use an ATmega328, but an Atmega8 is shown in the schematic for reference. The pin configuration is identical on all three processors.

Power

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.

- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication using the SPI library.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:

- **TWI: A4 or SDA pin and A5 or SCL pin.** Support TWI communication using the Wire library.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

See also the mapping between Arduino pins and ATmega328 ports. The mapping for the Atmega8, 168, and 328 is identical.

Communication

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Uno's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

Programming

The Arduino Uno can be programmed with the Arduino software (download). Select "Arduino Uno from the **Tools > Board** menu (according to the microcontroller on your board). For details, see the reference and tutorials.

The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available . The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

- On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.
- On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information.

Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Four screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

Share|
©Arduino | Edit Page | Page History | Printable View | All Recent Site Changes

Anexo E. Datasheet ESP8266

ESP8266EX
Datasheet

Version 1.0

Copyright © 2016

About This Guide

This document introduces the specifications of ESP8266EX, including the following topics.

Chapter	Title	Subject
Chapter 1	Overview	Provides overview introduction to ESP82 serials, including features, protocols, technical parameters and applications.
Chapter 2	Pin Definitions	Provides introduction to pin layout and the relevant description.
Chapter 3	Functional Description	Describes major functional modules and protocols applied on ESP8266EX including CPU, flash and memory, clock, radio, Wi-Fi, and low-power management.
Chapter 4	Peripheral Interface	Provides descriptions of peripheral interfaces integrated on ESP8266EX.
Chapter 5	Electrical Specifications	Lists the electrical data of ESP8266EX.
Chapter 6	Package Information	Illustrates the package details for ESP8266EX.
Appendix	Pin List	Provides detailed pin information including digital die pin list, buffer sheet, register list, and strapping pin list.

Release Notes

Date	Version	Release Notes
2015.12	V4.6	Chapter 3 updated.
2016.02	V4.7	Section 3.6 updated.
2016.03	V4.8	Section 4.1 updated.
2016.04	V4.9	Chapter 1 updated.

Table of Contents

1. Overview

Espressif's ESP8266EX delivers highly integrated Wi-Fi SoC solution to meet users' continuous demands for efficient power usage, compact design and reliable performance in the Internet of Things industry.

With the complete and self-contained Wi-Fi networking capabilities, ESP8266EX can perform either as a standalone application or as the slave to a host MCU. When ESP8266EX hosts the application, it promptly boots up from the flash. The integrated high-speed cache helps to increase the system performance and optimize the system memory. Also, ESP8266EX can be applied to any micro-controller design as a Wi-Fi adaptor through SPI / SDIO or I2C / UART interfaces.

ESP8266EX integrates antenna switches, RF balun, power amplifier, low noise receive amplifier, filters and power management modules. The compact design minimizes the PCB size and requires minimal external circuitries.

Besides the Wi-Fi functionalities, ESP8266EX also integrates an enhanced version of Tensilica's L106 Diamond series 32-bit processor and on-chip SRAM. It can be interfaced with external sensors and other devices through the GPIOs. Software Development Kit (SDK) provides sample codes for various applications.

Espressif Systems' Smart Connectivity Platform (ESCP) enables sophisticated features including fast switch between sleep and wake-up mode for energy-efficient purpose, adaptive radio biasing for low-power operation, advance signal processing, spur cancellation and radio co-existence mechanisms for common cellular, Bluetooth, DDR, LVDS, LCD interference mitigation.

1.1. Wi-Fi Protocols

- 802.11 b/g/n/e/i support.
- Wi-Fi Direct (P2P) support.
- P2P Discovery, P2P Group Owner mode, P2P Power Management.
- Infrastructure BSS Station mode / P2P mode / softAP mode support.
- Hardware accelerators for CCMP (CBC-MAC, counter mode), TKIP (MIC, RC4), WAPI (SMS4), WEP (RC4), CRC.
- WPA/WPA2 PSK, and WPS driver.
- Additional 802.11i security features such as pre-authentication, and TSN.
- Open Interface for various upper layer authentication schemes over EAP such as TLS, PEAP, LEAP, SIM, AKA, or customer specific.
- 802.11n support (2.4 GHz).
- Supports MIMO 1×1 and 2×1, STBC, A-MPDU and A-MSDU aggregation and 0.4μs guard interval.
- WMM power save U-APSD.

- Multiple queue management to fully utilize traffic prioritization defined by 802.11e standard.
- UMA compliant and certified.
- 802.1h/RFC1042 frame encapsulation.
- Scattered DMA for optimal CPU off load on Zero Copy data transfer operations.
- Antenna diversity and selection (software managed hardware).
- Clock/power gating combined with 802.11-compliant power management dynamically adapted to current connection condition providing minimal power consumption.
- Adaptive rate fallback algorithm sets the optimum transmission rate and Tx power based on actual SNR and packet loss information.
- Automatic retransmission and response on MAC to avoid packet discarding on slow host environment.
- Seamless roaming support.
- Configurable packet traffic arbitration (PTA) with dedicated slave processor based design provides flexible and exact timing Bluetooth co-existence support for a wide range of Bluetooth Chip vendors.
- Dual and single antenna Bluetooth co-existence support with optional simultaneous receive (Wi-Fi/Bluetooth) capability.

1.2. Main Technical Specifications

Table 1-1. Main Technical Specifications

Categories	Items	Parameters
Wi-Fi	Standards	FCC/CE/TELEC/SRRC
	Protocols	802.11 b/g/n/e/i
	Frequency Range	2.4 G ~ 2.5 G (2400M ~ 2483.5M)
	Tx Power	802.11 b: +20 dBm
		802.11 g: +17 dBm
		802.11 n: +14 dBm
	Rx Sensitivity	802.11 b: -91 dbm (11 Mbps)
		802.11 g: -75 dbm (54 Mbps)
		802.11 n: -72 dbm (MCS7)
	Antenna	PCB Trace, External, IPEX Connector, Ceramic Chip
Hardware	CPU	Tensilica L106 32-bit micro controller
	Peripheral Interface	UART/SDIO/SPI/I2C/I2S/IR Remote Control
		GPIO/ADC/PWM
	Operating Voltage	3.0 V ~ 3.6 V
	Operating Current	Average value: 80 mA
	Operating Temperature Range	-40°C ~ 125°C
	Storage Temperature Range	-40°C ~ 125°C
	Package Size	QFN32-pin (5 mm x 5 mm)
	External Interface	-
Software	Wi-Fi Mode	station/softAP/SoftAP+station
	Security	WPA/WPA2
	Encryption	WEP/TKIP/AES
	Firmware Upgrade	UART Download / OTA (via network)
	Software Development	Supports Cloud Server Development / Firmware and SDK for fast on-chip programming
	Network Protocols	IPv4, TCP/UDP/HTTP/FTP
	User Configuration	AT Instruction Set, Cloud Server, Android/iOS App

1.3. Applications

- Home Appliances
- Home Automation
- Smart Plugs and Lights
- Mesh Network

- Industrial Wireless Control
- Baby Monitors
- IP Cameras
- Sensor Networks
- Wearable Electronics
- Wi-Fi Location-aware Devices
- Security ID Tags
- Wi-Fi Position System Beacons

2. Pin Definitions

Figure 2-1 shows the pin layout for 32-pin QFN package.

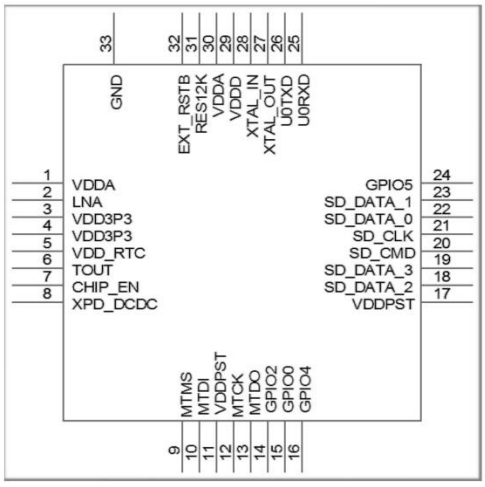


Figure 2-1. Pin Layout

Table 2-1 lists the definitions and functions of each pin.

Table 2-1. ESP8266EX Pin Definitions

Pin	Name	Type	Function
1	VDDA	P	Analog Power 3.0 V ~ 3.6 V
2	LNA	I/O	RF Antenna Interface. Chip Output Impedance=50 Ω No matching required. It is suggested to retain the π-type matching network to match the antenna.
3	VDD3P3	P	Amplifier Power 3.0 V ~ 3.6 V
4	VDD3P3	P	Amplifier Power 3.0 V ~ 3.6 V
5	VDD_RTC	P	NC (1.1 V)
6	TOUT	I	ADC pin. It can be used to test the power-supply voltage of VDD3P3 (Pin3 and Pin4) and the input power voltage of TOUT (Pin 6). However, these two functions cannot be used simultaneously.
7	CHIP_PU	I	Chip Enable High: On, chip works properly Low: Off, small current consumed
8	XPD_DCDC	I/O	Deep-sleep wakeup (need to be connected to EXT_RSTB); GPIO16

Pin	Name	Type	Function
9	MTMS	I/O	GPIO14; HSPI_CLK
10	MTDI	I/O	GPIO12; HSPI_MISO
11	VDDPST	P	Digital/IO Power Supply (1.8 V ~ 3.3 V)
12	MTCK	I/O	GPIO13; HSPI_MOSI; UART0_CTS
13	MTDO	I/O	GPIO15; HSPI_CS; UART0_RTS
14	GPIO2	I/O	UART Tx during flash programming; GPIO2
15	GPIO0	I/O	GPIO0; SPI_CS2
16	GPIO4	I/O	GPIO4
17	VDDPST	P	Digital/IO Power Supply (1.8 V ~ 3.3 V)
18	SDIO_DATA_2	I/O	Connect to SD_D2 (Series R: 200Ω); SPIHD; HSPIHD; GPIO9
19	SDIO_DATA_3	I/O	Connect to SD_D3 (Series R: 200Ω); SPIWP; HSPIWP; GPIO10
20	SDIO_CMD	I/O	Connect to SD_CMD (Series R: 200Ω); SPI_CS0; GPIO11
21	SDIO_CLK	I/O	Connect to SD_CLK (Series R: 200Ω); SPI_CLK; GPIO6
22	SDIO_DATA_0	I/O	Connect to SD_D0 (Series R: 200Ω); SPI_MSIO; GPIO7
23	SDIO_DATA_1	I/O	Connect to SD_D1 (Series R: 200Ω); SPI_MOSI; GPIO8
24	GPIO5	I/O	GPIO5
25	U0RXD	I/O	UART Rx during flash programming; GPIO3
26	U0TXD	I/O	UART Tx during flash progamming; GPIO1; SPI_CS1
27	XTAL_OUT	I/O	Connect to crystal oscillator output, can be used to provide BT clock input
28	XTAL_IN	I/O	Connect to crystal oscillator input
29	VDDD	P	Analog Power 3.0 V ~ 3.6 V
30	VDDA	P	Analog Power 3.0 V ~ 3.6 V
31	RES12K	I	Serial connection with a 12 kΩ resistor and connect to the ground
32	EXT_RSTB	I	External reset signal (Low voltage level: Active)

Note:

GPIO2, GPIO0, and MTDO are configurable on PCB as the 3-bit strapping register that determines the booting mode and the SDIO timing mode.

3. Functional Description

The functional diagram of ESP8266EX is shown as in Figure 3-1.

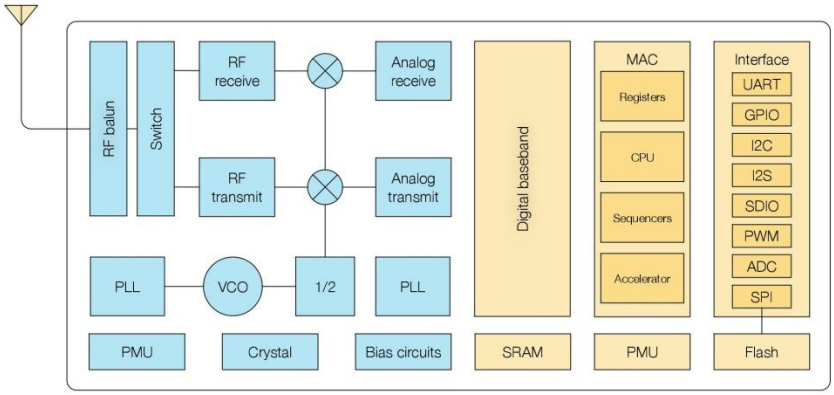


Figure 3-1. Functional Block Diagram

3.1. CPU, Memory, and Flash

3.1.1. CPU

ESP8266EX integrates Tensilica L106 32-bit micro controller (MCU) and ultra-low-power 16-bit RSIC. The CPU clock speed is 80 MHz. It can also reach a maximum value of 160 MHz. Real Time Operation System (RTOS) is enabled. Currently, only 20% of MIPS has been occupied by the Wi-Fi stack, the rest can all be used for user application programming and development. The CPU includes the interfaces as below.

- Programmable RAM/ROM interfaces (iBus), which can be connected with memory controller, and can also be used to visit flash.
- Data RAM interface (dBus), which can connected with memory controller.
- AHB interface which can be used to visit the register.

3.1.2. Memory

ESP8266EX Wi-Fi SoC integrates memory controller and memory units including SRAM and ROM. MCU can access the memory units through iBus, dBus, and AHB interfaces. All memory units can be accessed upon request, while a memory arbiter will decide the running sequence according to the time when these requests are received by the processor.

According to our current version of SDK, SRAM space available to users is assigned as below.

- RAM size < 50 kB, that is, when ESP8266EX is working under the station mode and connects to the router, programmable space accessible in heap + data section is around 50kB.
- There is no programmable ROM in the SoC, therefore, user program must be stored in an external SPI flash.

3.1.3. External Flash

ESP8266EX uses external SPI flash to store user programs, and supports up to 16 Mbytes memory capacity theoretically.

The minimum flash memory of ESP8266EX is shown in Table 3-1.

Table 3-1. Minimum Flash Memory

OTA	Minimum Flash Memory
disabled	512 kB
enabled	1 MB

3.2. AHB and AHB Blocks

The AHB block performs as an arbiter. It controls the AHB interfaces through the MAC, SDIO (host) and CPU. Depending on the address, the AHB data requests can go into one of the two slaves.

- APB block
- Flash controller (usually for standalone applications)

Data requests to the memory controller are usually high speed requests, and requests to the APB block are usually register access.

The APB block acts as a decoder that only accesses the programmable registers within the main blocks of ESP8266EX. Depending on the address, the APB request can go to radio, SI/SPI, SDIO (host), GPIO, UART, real-time clock (RTC), MAC or digital baseband.

3.3. Clock

3.3.1. High Frequency Clock

The high frequency clock on ESP8266EX is used to drive both transmit and receive mixers. This clock is generated from internal crystal oscillator and external crystal. The crystal frequency ranges from 26 MHz to 52 MHz.

The internal calibration inside the crystal oscillator ensures that a wide range of crystals can be used, nevertheless the quality of the crystal is still a factor to consider to have reasonable phase noise and good Wi-Fi sensitivity. Refer to Table 3-2 to measure the frequency offset.

Table 3-2. High Frequency Clock Specifications

Parameter	Symbol	Min	Max	Unit
Frequency	FXO	26	52	MHz
Loading capacitance	CL	–	32	pF
Motional capacitance	CM	2	5	pF
Series resistance	RS	0	65	Ω
Frequency tolerance	ΔFXO	-15	15	ppm
Frequency vs temperature (-25°C ~ 75°C)	ΔFXO,Temp	-15	15	ppm

3.3.2. External Clock Requirements

An externally generated clock is available with the frequency ranging from 26 MHz to 52 MHz. The following characteristics are expected to achieve good performance of radio.

Table 3-3. External Clock Reference

Parameter	Symbol	Min	Max	Unit
Clock amplitude	VXO	0.2	1	Vpp
External clock accuracy	ΔFXO,EXT	-15	15	ppm
Phase noise @1kHz offset, 40 MHz clock	–	–	-120	dBc/Hz
Phase noise @10kHz offset, 40 MHz clock	–	–	-130	dBc/Hz
Phase noise @100kHz offset, 40 MHz clock	–	–	-138	dBc/Hz

3.4. Radio

ESP8266EX radio consists of the following blocks.

- 2.4 GHz receiver
- 2.4 GHz transmitter
- High speed clock generators and crystal oscillator
- Real time clock
- Bias and regulators
- Power management

3.4.1. Channel Frequencies

The RF transceiver supports the following channels according to IEEE802.11b/g/n standards.

Table 3-4. Frequency Channel

Channel No.	Frequency (MHz)	Channel No.	Frequency (MHz)
1	2412	8	2447
2	2417	9	2452
3	2422	10	2457
4	2427	11	2462
5	2432	12	2467
6	2437	13	2472
7	2442	14	2484

3.4.2. 2.4 GHz Receiver

The 2.4 GHz receiver down-converts the RF signals to quadrature baseband signals and converts them to the digital domain with 2 high resolution high speed ADCs. To adapt to varying signal channel conditions, RF filters, automatic gain control (AGC), DC offset cancelation circuits and baseband filters are integrated within ESP8266EX.

3.4.3. 2.4 GHz Transmitter

The 2.4 GHz transmitter up-converts the quadrature baseband signals to 2.4 GHz, and drives the antenna with a high-power CMOS power amplifier. The function of digital calibration further improves the linearity of the power amplifier, enabling a state of art performance of delivering +19.5 dBm average power for 802.11b transmission and +16 dBm for 802.11n transmission.

Additional calibrations are integrated to offset any imperfections of the radio, such as:

- Carrier leakage
- I/Q phase matching
- Baseband nonlinearities

These built-in calibration functions reduce the product test time and make the test equipment unnecessary.

3.4.4. Clock Generator

The clock generator generates quadrature 2.4 GHz clock signals for the receiver and transmitter. All components of the clock generator are integrated on the chip, including all inductors, varactors, filters, regulators and dividers.

The clock generator has built-in calibration and self test circuits. Quadrature clock phases and phase noise are optimized on-chip with patented calibration algorithms to ensure the best performance of the receiver and transmitter.

3.5. Wi-Fi

ESP8266EX implements TCP/IP, the full 802.11 b/g/n/e/i WLAN MAC protocol and Wi-Fi Direct specification. It supports not only basic service set (BSS) operations under the distributed control function (DCF) but also P2P group operation compliant with the latest Wi-Fi P2P protocol. Low level protocol functions are handled automatically by ESP8266EX.

- RTS/CTS
- acknowledgement
- fragmentation and defragmentation
- aggregation
- frame encapsulation (802.11h/RFC 1042)
- automatic beacon monitoring / scanning, and
- P2P Wi-Fi direct

Passive or active scanning, as well as P2P discovery procedure is performed autonomously once initiated by the appropriate command. Power management is handled with minimum interaction with host to minimize active duty period.

3.6. Power Management

ESP8266EX is dedicated designed for mobile devices, wearable electronics and the Internet of Things applications with advanced power management technologies.

The low-power architecture operates in 3 modes: active mode, sleep mode and deep sleep mode. ESP8266EX consumes about than 20μA in deep sleep mode (with RTC clock still running) and less than 1.0mA (DTIM=3) or less than 0.6mA (DTIM=10) to stay connected to the access point.

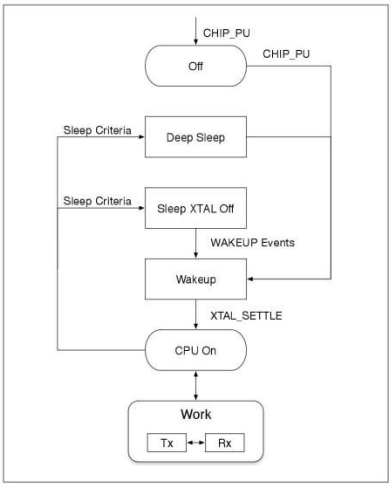


Figure 3-2. Power Management

- **Off**: CHIP_PU pin is low. The RTC is disabled. All registers are cleared.
- **Deep sleep**: Only RTC is powered on – the rest of the chip is powered off. Recovery memory of RTC can keep basic Wi-Fi connecting information.
- **Sleep**: Only the RTC is operating. The crystal oscillator is disabled. Any wake-up events (MAC, host, RTC timer, external interrupts) will put the chip into the wake up mode.
- **Wake up**: In this state, the system switches from the sleep states to the PWR mode. The crystal oscillator and PLLs are enabled.
- **On**: The high speed clock is operational and sent to each block enabled by the clock control register. Lower level clock gating is implemented at the block level, including the CPU, which can be gated off using the WAITI instruction while the system is on.

***Note*:**

For more information about sleep mode, refer to ***ESP8266 Sleep Function Description***.

4. Peripheral Interface

4.1. General Purpose Input/Output Interface (GPIO)

ESP8266EX has 17 GPIO pins which can be assigned to various functions by programming the appropriate registers.

Each GPIO can be configured with internal pull-up or pull-down, or set to high impedance, and when configured as an input, the data are stored in software registers; the input can also be set to edge-trigger or level trigger CPU interrupts. In short, the IO pads are bi-directional, non-inverting and tristate, which includes input and output buffer with tristate control inputs.

These pins can be multiplexed with other functions such as I2C, I2S, UART, PWM, IR Remote Control, etc.

For low power operations, the GPIOs can also be set to hold their state. For instance, when the chip is powered down, all output enable signals can be set to hold low.

Optional hold functionality can be built into the IO if requested. When the IO is not driven by the internal or external circuitry, the hold functionality can be used to hold the state to the last used state. The hold functionality introduces some positive feedback into the pad. Hence, the external driver that drives the pad must be stronger than the positive feedback. The required drive strength is small — in the range of 5μA to pull apart the latch.

4.2. Secure Digital Input/Output Interface (SDIO)

ESP8266EX has one Slave SDIO, the definitions of which are described as Table 4-1.

Table 4-1. Pin Definitions of SDIOs

Pin Name	Pin Num	IO	Function Name
SDIO_CLK	21	IO6	SDIO_CLK
SDIO_DATA0	22	IO7	SDIO_DATA0
SDIO_DATA1	23	IO8	SDIO_DATA1
SDIO_DATA_2	18	IO9	SDIO_DATA_2
SDIO_DATA_3	19	IO10	SDIO_DATA_3
SDIO_CMD	20	IO11	SDIO_CMD

Note:

4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported.

4.3. Serial Peripheral Interface (SPI/HSPI)

ESP8266EX has 3 SPIs.

- One general Slave/Master SPI
- One Slave SDIO/SPI
- One general Slave/Master HSPI

Functions of all these pins can be implemented via hardware. The pin definitions are described as below.

4.3.1. General SPI (Master/Slave)

Table 4-2. Pin Definitions of SPIs

Pin Name	Pin Num	IO	Function Name
SDIO_CLK	21	IO6	SPICLK
SDIO_DATA0	22	IO7	SPIQ/MISO
SDIO_DATA1	23	IO8	SPID/MOSI
SDIO_DATA_2	18	IO9	SPIHD
SDIO_DATA_3	19	IO10	SPIWP
U0TXD	26	IO1	SPICS1
GPIO0	15	IO0	SPICS2

Note:
SPI mode can be implemented via software programming. The clock frequency is 80 MHz at maximum.

4.3.2. HSPI (Slave)

Table 4-3. Pin Definitions of HSPI (Slave)

Pin Name	Pin Num	IO	Function Name
MTMS	9	IO14	HSPICLK
MTDI	10	IO12	HSPIQ/MISO
MTCK	12	IO13	HSPID/MOSI
MTDO	13	IO15	HPSICS

4.4. I2C Interface

ESP8266EX has one I2C used to connect with micro-controller and other peripheral equipments such as sensors. The pin definition of I2C is as below.

Table 4-4. Pin Definitions of I2C

Pin Name	Pin Num	IO	Function Name
MTMS	9	IO14	I2C_SCL
GPIO2	14	IO2	I2C_SDA

Both I2C Master and I2C Slave are supported. I2C interface functionality can be realized via software programming, the clock frequency reaches 100 kHz at a maximum. It should be noted that I2C clock frequency should be higher than the slowest clock frequency of the slave device.

4.5. I2S Interface

ESP8266EX has one I2S data input interface and one I2S data output interface. I2S interfaces are mainly used in applications such as data collection, processing, and transmission of audio data, as well as the input and output of serial data. For example, LED lights (WS2812 series) are supported. The pin definition of I2S is as below. I2S functionality can be enabled via software programming by using multiplexed GPIOs, and linked list DMA is supported.

Table 4-5. Pin Definitions of I2S

I2S Data Input			
Pin Name	Pin Num	IO	Function Name
MTDI	10	IO12	I2SI_DATA
MTCK	12	IO13	I2SI_BCK
MTMS	9	IO14	I2SI_WS
MTDO	13	IO15	I2SO_BCK
U0RXD	25	IO3	I2SO_DATA
GPIO2	14	IO2	I2SO_WS

4.6. Universal Asynchronous Receiver Transmitter (UART)

ESP8266EX has two UART interfaces UART0 and UART, the definitions are as below.

Table 4-6. Pin Definitions of UART

Pin Type	Pin Name	Pin Num	IO	Function Name
UART0	U0RXD	25	IO3	U0RXD
	U0TXD	26	IO1	U0TXD
	MTDO	13	IO15	U0RTS
	MTCK	12	IO13	U0CTS

Pin Type	Pin Name	Pin Num	IO	Function Name
UART1	GPIO2	14	IO2	U1TXD
	SD_D1	23	IO8	U1RXD

Data transfers to/from UART interfaces can be implemented via hardware. The data transmission speed via UART interfaces reaches 115200 x 40 (4.5 Mbps).

UART0 can be used for communication. It supports fluid control. Since UART1 features only data transmit signal (Tx), it is usually used for printing log.

Note:

By default, UART0 outputs some printed information when the device is powered on and booting up. The baud rate of the printed information is relevant to the frequency of the external crystal oscillator. If the frequency of the crystal oscillator is 40 MHz, then the baud rate for printing is 115200; if the frequency of the crystal oscillator is 26 MHz, then the baud rate for printing is 74880. If the printed information exerts any influence on the functionality of the device, it is suggested to block the printing during the power-on period by changing (U0TXD,U0RXD) to (MTDO,MTCK).

4.7. Pulse-Width Modulation (PWM)

ESP8266EX has four PWM output interfaces. They can be extended by users themselves. The pin definitions of the PWM interfaces are defined as below.

Table 4-7. Pin Definitions of PWM

Pin Name	Pin Num	IO	Function Name
MTDI	10	IO12	PWM0
MTDO	13	IO15	PWM1
MTMS	9	IO14	PWM2
GPIO4	16	IO4	PWM3

The functionality of PWM interfaces can be implemented via software programming. For example, in the LED smart light demo, the function of PWM is realized by interruption of the timer, the minimum resolution reaches as much as 44 ns. PWM frequency range is adjustable from 1000 μs to 10000 μs, i.e., between 100Hz and 1 kHz. When the PWM frequency is 1 kHz, the duty ratio will be 1/22727, and over 14 bit resolution will be achieved at 1 kHz refresh rate.

4.8. IR Remote Control

One Infrared remote control interface is defined as below.

Table 4-8. Pin Definitions of IR Remote Control

Pin Name	Pin Num	IO	Function Name
MTMS	9	IO14	IR Tx

Pin Name	Pin Num	IO	Function Name
GPIO5	24	IO5	IR Rx

The functionality of Infrared remote control interface can be implemented via software programming. NEC coding, modulation, and demodulation are used by this interface. The frequency of modulated carrier signal is 38 kHz, while the duty ratio of the square wave is 1/3. The transmission range is around 1m which is determined by two factors: one is the maximum value of rated current, the other is internal current-limiting resistance value in the infrared receiver. The larger the resistance value, the lower the current, so is the power, and vice versa. The transmission angle is between 15° and 30° which is determined by the radiation direction of the infrared receiver.

4.9. ADC (Analog-to-Digital Converter)

ESP8266EX is embedded with a 10-bit precision SARADC. TOUT (Pin6) is defined as below.

Table 4-9. Pin Definition of ADC

Pin Name	Pin Num	Function Name
TOUT	6	ADC Interface

The following two functions can be implemented using ADC (Pin 6). However, they cannot be implemented at the same time.

- Test the power supply voltage of VDD3P3 (Pin 3 and Pin 4).

Hardware Design	TOUT must be dangled.
RF Initialization Parameter	The 107th byte of ***esp_init_data_default.bin*** (0 ~ 127 bytes), `vdd33_const` must be set to `0xFF`.
RF Calibration Process	Optimize the RF circuit conditions based on the testing results of VDD3P3 (Pin 3 and Pin 4).
User Programming	Use `system_get_vdd33` instead of `system_adc_read`.

- Test the input voltage of TOUT (Pin 6).

Hardware Design	The input voltage range is 0 to 1.0V when TOUT is connected to external circuit.
RF Initialization Parameter	The value of the 107th byte of ***esp_init_data_default.bin*** (0 ~ 127 bytes), `vdd33_const` must be set to the real power supply voltage of Pin 3 and Pin 4. The working power voltage range of ESP8266EX is between 1.8V and 3.6V, while the unit of `vdd33_const` is 0.1V, therefore, the effective value range of `vdd33_const` is 18 to 36.
RF Calibration Process	Optimize the RF circuit conditions based on the value of `vdd33_const`. The permissible error is ±0.2V.
User Programming	Use `system_adc_read` instead of `system_get_vdd33`.

Notes:

esp_init_data_default.bin *is provided in SDK package which contains RF initialization parameters (0 ~ 127 bytes).*

You can define the 107th byte in ***esp_init_data_default.bin*** *to* `vdd33_const` *as below.*

- *If vdd33_const = 0xff, the power voltage of Pin 3 and Pin 4 will be tested by the internal self-calibration process of ESP8266EX itself. RF circuit conditions should be optimized according to the testing results.*
- *If 18 =< vdd33_const =< 36, ESP8266EX RF Calibration and optimization process is implemented via (vdd33_const/10).*
- *If vdd33_const < 18 or 36 < vdd33_const < 255, ESP8266EX RF Calibration and optimization process is implemented via the default value 3.0V.*

4.10. LED Light and Button

ESP8266EX features 17 GPIOs, all of which can be assigned to support various functions of LED lights and buttons. Definitions of some GPIOs that are assigned with certain functions in demo application design are shown as below.

Table 4-10. Pin Definition of LED and Button

Pin Name	Pin Num	IO	Function Name
MTCK	12	IO13	Button (Reset)
GPIO0	15	IO0	Wi-Fi Light
MTDI	10	IO12	Link Light

Altogether three interfaces have been defined, one is for the button, while the other two are for LED light. Generally, MTCK is used to control the reset button, GPIO0 is used as an signal to indicate the Wi-Fi working state, MTDI is used as a signal light to indicate communication status between the device and the server.

Note:

Most interfaces described in this chapter can be multiplexed. Pin definitions that can be defined is not limited to the ones herein mentioned, you can customize the functions of the pins according to your specific application scenarios via software programming and hardware design.

5. Electrical Specifications

5.1. Electrical Characteristics

Table 5-1. Electrical Characteristics

Parameters		Conditions	Min	Typical	Max	Unit
Storage Temperature Range		-	-40	Normal	125	°C
Maximum Soldering Temperature		IPC/JEDEC J-STD-020	-	-	260	°C
Working Voltage Value		—	3.0	3.3	3.6	V
I/O	V_{IL}/V_{IH}	-	-0.3/0.75V_{IO}	-	0.25V_{IO}/3.6	V
	V_{OL}/V_{OH}	-	N/0.8V_{IO}	-	0.1V_{IO}/N	
	I_{MAX}	-	-	-	12	mA
Electrostatic Discharge (HBM)		TAMB=25°C	-	-	2	KV
Electrostatic Discharge (CDM)		TAMB=25°C	-	-	0.5	KV

5.2. Power Consumption

Table 5-2. Power Consumption

Parameters	Min	Typical	Max	Unit
Tx802.11b, CCK 11Mbps, P OUT=+17dBm	-	170	-	mA
Tx 802.11g, OFDM 54Mbps, P OUT =+15dBm	-	140	-	mA
Tx 802.11n, MCS7, P OUT =+13dBm	-	120	-	mA
Rx 802.11b, 1024 bytes packet length , -80dBm	-	50	-	mA
Rx 802.11g, 1024 bytes packet length, -70dBm	-	56	-	mA
Rx 802.11n, 1024 bytes packet length, -65dBm	-	56	-	mA
Modem-Sleep①	-	15	-	mA
Light-Sleep②	-	0.9	-	mA
Deep-Sleep③	-	10	-	µA
Power Off	-	0.5	-	µA

Notes:

- *Modem-Sleep mode is used in the applications that require the CPU to be working, as in PWM or I2S applications. According to 802.11 standards (like U-APSD), it shuts down the Wi-Fi Modem circuit while maintaining a Wi-Fi connection with no data transmission to optimize power consumption. E.g. in DTIM3, maintaining a sleep of 300ms with a wake-up of 3ms cycle to receive AP's Beacon packages at interval requires about 15mA current.*
- *During Light-Sleep mode, the CPU may be suspended in applications like Wi-Fi switch. Without data transmission, the Wi-Fi Modem circuit can be turned off and CPU suspended to save power consumption according to the 802.11 standards (U-APSD). E.g. in DTIM3, maintaining a sleep of 300ms with a wake-up of 3ms to receive AP's Beacon packages at interval requires about 0.9mA current.*
- *During Deep-Sleep mode, Wi-Fi is turned off. For applications with long time lags between data transmission, e.g. a temperature sensor that detects the temperature every 100s, sleeps for 300s and wakes up to connect to the AP (taking about 0.3 ~ 1s), the overall average current is less than 1mA.*

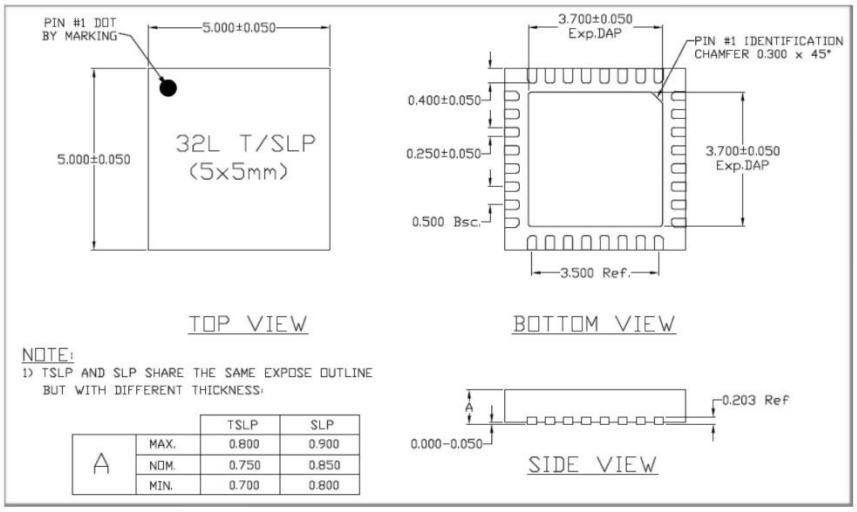
5.3. Wi-Fi Radio Characteristics

The following data are from tests conducted at room temperature with 3.3V and 1.1V power supplies.

Table 5-3. Wi-Fi Radio Characteristics

Parameters	Min	Typical	Max	Unit
Input frequency	2412	-	2484	MHz
Input impedance	-	50	-	Ω
Input reflection	-	-	-10	dB
Output power of PA for 72.2 Mbps	15.5	16.5	17.5	dBm
Output power of PA for 11b mode	19.5	20.5	21.5	dBm
Sensitivity	-	-	-	–
DSSS, 1 Mbps	-	-98	-	dBm
CCK, 11 Mbps	-	-91	-	dBm
6 Mbps (1/2 BPSK)	-	-93	-	dBm
54 Mbps (3/4 64-QAM)	-	-75	-	dBm
HT20, MCS7 (65 Mbps, 72.2 Mbps)	-	-72	-	dBm
Adjacent Channel Rejection				
OFDM, 6 Mbps	-	37	-	dB
OFDM, 54 Mbps	-	21	-	dB
HT20, MCS0	-	37	-	dB
HT20, MCS7	-	20	-	dB

6. Package Information



		TSLP	SLP
A	MAX.	0.800	0.900
	NOM.	0.750	0.850
	MIN.	0.700	0.800

Figure 6-1. ESP8266EX Package

I. Appendix

For detailed pin information, refer to ***ESP8266EX Pin List***.

- Digital Die Pin List
- Buffer Sheet
- Register List
- Strapping List

Notes:

- *INST_NAME refers to the IO_MUX REGISTER defined in eagle_soc.h, for example MTDI_U refers to PERIPHS_IO_MUX_MTDI_U.*
- *Net Name refers to the pin name in schematic.*
- *Function refers to the multifunction of each pin pad.*
- *Function number 1 ~ 5 correspond to FUNCTION 0 ~ 4 in SDK. For example, set MTDI to GPIO12 as follows.*

```
- #define FUNC_GPIO12  3 //defined in eagle_soc.h
- PIN_FUNC_SELECT(PERIPHS_IO_MUX_MTDI_U,FUNC_GPIO12)
```

Espressif IOT Team

www.espressif.com

Disclaimer and Copyright Notice

Information in this document, including URL references, is subject to change without notice.

THIS DOCUMENT IS PROVIDED AS IS WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE.

All liability, including liability for infringement of any proprietary rights, relating to use of information in this document is disclaimed. No licenses express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

The Wi-Fi Alliance Member logo is a trademark of the Wi-Fi Alliance. The Bluetooth logo is a registered trademark of Bluetooth SIG.

All trade names, trademarks and registered trademarks mentioned in this document are property of their respective owners, and are hereby acknowledged.

Copyright © 2016 Espressif Inc. All rights reserved.

Anexo F. Código en Arduino del prototipo - comentarios en color naranja.

```
//Librerías para el uso de los sensores
#include <Wire.h> //Librería para comunicacin con dispositivos I2C
#include <SoftwareSerial.h> //Librería para la comunicación del Arduino por el puerto serie
#include <stdlib.h> //Librería estándar lenguaje C
#include <NewPing.h> //Librería para manejo ultrasonido
//Definición de constantes
#define DEVICE (0x53)    //Dirección del dispositivo ADXL345
#define TO_READ (6)     //Número de bytes que se usaran para leer los ejes del ADXL345 (2 bytes por eje)
#define TRIGGER_PIN    7     // Pin de Arduino conectado al pin Trigger, en el sensor HC-SR04
#define ECHO_PIN       6     // Pin de Arduino conectado al pin Echo, en el sensor HC-SR04.
#define TRIGGER_PIN_2    5     // Pin de Arduino conectado al pin Trigger, en el segundo sensor HC-SR04.
#define ECHO_PIN_2       4     // Pin de Arduino conectado al pin Echo, en el segundo sensor HC-SR04.
#define MAX_DISTANCE 100     // Distancia máxima que podrá medir el sensor HC-SR04.
#define SOUND_SPEED 0.171    // La mitad de la velocidad del sonido en el aire, medida en [mm/us]
//Otras variables globales, configuración de librerías y envío de información a ThingSpeak
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE); // Configuración de la librería NewPing
NewPing sonar_2(TRIGGER_PIN_2, ECHO_PIN_2, MAX_DISTANCE); // Configuración de la librería NewPing
byte buff[TO_READ] ;      //buffer de 6 bytes para guardar los datos leídos por el ADXL345
char str[512];           //buffer de tipo string para transformar los datos antes de enviarlos al monitor serie
int regAddress = 0x32;    //primer registro de aceleracion en el ADXL345
int x, y, z;                  //datos de los tres ejes de aceleración
double roll = 0.00, pitch = 0.00;     //Ángulos en los cuales rota el ADXL345 en los ejes X y Y (Roll & Pitch)
String apiKey = "BYIO25TYJ1OVGMUP"; //Variables tipo string con la llave de escritura del canal en Thingspeak
SoftwareSerial ser(10, 11); // Establece los pines para RX, TX en el Arduino para el ESP8266

//Rutina de ejecución única para alistar los dispositivos
void setup() {
  Wire.begin();       // Unirse al bus i2c
  Serial.begin(9600);  // Inicia el puerto serie para salida a 9600 baud

  //Encendido del ADXL345
  writeTo(DEVICE, 0x2D, 0);
  writeTo(DEVICE, 0x2D, 16);
  writeTo(DEVICE, 0x2D, 8);
  ser.begin(9600); //Comunicación del puerto serie a 9600 baud
```

```
  ser.println("AT+RST");   //Comando AT para reinicio del ESP8266
  delay(10000); //Espera de 10 segundos para que el ESP8266 termine el reinicio
}

//Código que se ejecuta en bucle
void loop() {

// Obtener medición de tiempo de viaje del sonido y guardar en variable uS
  int uS = sonar.ping_median();
  // Imprimir la distancia medida a la consola serial
  Serial.print("Distancia sensor ultrasonico columna: ");
  // Calcular la distancia con base en una constante
  Serial.print(uS / US_ROUNDTRIP_CM);
  int distance = (uS / US_ROUNDTRIP_CM);
  Serial.println("cm");

// Obtener medición de tiempo de viaje del sonido y guardar en variable uS
  int uS_2 = sonar_2.ping_median();
  // Imprimir la distancia medida a la consola serial
  Serial.print("Distancia sensor ultrasonico viga: ");
  // Calcular la distancia con base en una constante
  Serial.print(uS_2 / US_ROUNDTRIP_CM);
  int distance_2 = (uS_2 / US_ROUNDTRIP_CM);
  Serial.println("cm");

 readFrom(DEVICE, regAddress, TO_READ, buff); //lee la aceleración desde el ADXL345
                                //cada lectura entrante viene con 10 bit de resolucion, ie 2
bytes. El bit menos significativo primero.
                                //así se convierten ambos bytes en un entero
  x = (((int)buff[1]) << 8) | buff[0];  //asignación del valor de x
  y = (((int)buff[3])<< 8) | buff[2];   //asignación del valor de y
  z = (((int)buff[5]) << 8) | buff[4];  //asignación del valor de z

  //Envío de los valores x,y,z al puerto serie para la visualización en el monitor
  Serial.print("La aceleracion presente en x, y, z es:");
  sprintf(str, "%d %d %d", x, y, z);
  Serial.print(str);
  Serial.write(10);
  //Envio de los valores roll & pitch al puerto serie para la visualizacion en el monitor
  RP_calculate();
  Serial.print("Roll:"); Serial.println( roll );
  Serial.print("Pitch:"); Serial.println( pitch );
  Serial.println("");


//Envío de la información al canal en ThingSpeak.
//Convertir la información de tipo entero a string por medio de la función dtosrf.
  char bufX[16];                  //Buffer para convertir la aceleración en X
  String X = dtostrf(x,0,2,bufX);    //Conversión de la aceleración en X a String
```

```
  char bufY[16];                  //Buffer para convertir la aceleración en Y
  String Y = dtostrf(y,0,2,bufY);    //Conversión de la aceleración en Y a String
  char bufZ[16];                  //Buffer para convertir la aceleración en Z
  String Z = dtostrf(z,0,2,bufZ);    //Conversión de la aceleración en Z a String
  char bufR[16];                  //Buffer para convertir el ángulo Roll
  String R = dtostrf(roll,0,2,bufR); //Conversión del ángulo Roll a String
  char bufP[16];                  //Buffer para convertir el ángulo Pitch
  String P = dtostrf(pitch,0,2,bufP); //Conversión del ángulo Pitch a String
  char bufD[16];                  //Buffer para convertir la distancia
  String D = dtostrf(distance,0,2,bufD); //Conversión del ángulo Roll a String
  char bufD2[4];                  //Buffer para convertir la distancia 2
  String D2 = dtostrf(distance_2,0,2,bufD2); //Conversión de la distancia a String

//Establecimiento de la conexión TCP
  String cmd = "AT+CIPSTART=\"TCP\",\""; //Parte 1 del comando para establecer la
conexión TCP
  cmd += "184.106.153.149"; //Parte 2  del comando - Dirección IP pública ThingSpeak
  cmd += "\",80";  //Parte 3 del comando - puerto para la conexión TCP
  ser.println(cmd); //Concatenar las tres partes del comando de conexión y enviarlo al
ESP8266
  Serial.println(cmd); //Impresión en el monitor serie del comando enviado
//Condicional para verificar que se estableció la conexión correctamente
  if(ser.find("Error")){
    Serial.println("AT+CIPSTART error");
    return;
  }
  //Preparar cadena de caracteres GET
  String getStr = "GET /update?api_key="; //Parte 1 de la cadena de caracteres a enviar
  getStr += apiKey;  //Parte 2 de la cadena de caracteres a enviar - llave escritura
ThingSpeak
  getStr +="&field1=";  //Parte 3 de la cadena de caracteres a enviar - valor del campo 1
del canal
  getStr += String(X);
  getStr +="&field2=";  //Parte 4 de la cadena de caracteres a enviar - valor del campo 2
del canal
  getStr += String(Y);
  getStr +="&field3=";  //Parte 5 de la cadena de caracteres a enviar - valor del campo 3
del canal
  getStr += String(Z);
  getStr +="&field4=";  //Parte 6 de la cadena de caracteres a enviar - valor del campo 4
del canal
  getStr += String(R);
  getStr +="&field5=";  //Parte 7 de la cadena de caracteres a enviar - valor del campo 5
del canal
  getStr += String(P);
  getStr +="&field6=";  //Parte 8 de la cadena de caracteres a enviar - valor del campo 6
del canal
  getStr += String(D);
```

```
getStr +="&field7=";  //Parte 9 de la cadena de caracteres a enviar - valor del campo 7 del canal
  getStr += String(D2);
  getStr += "\r\n";
//Enviar el tamaño de la cadena
  cmd = "AT+CIPSEND=";  //Comando AT previo al envío de la cadena de caracteres
  cmd += String(getStr.length());  //Obtención de la longitud de la cadena GET completa
  ser.println(cmd); //envío del comando AT+CIPSEND con la longitud de la cadena al canal en ThingSpeak
  Serial.println(cmd); //Visualización del comando enviado en el monitor serie
  if(ser.find(">")){  //Condicional para comprobar que fue enviada la longitud de la cadena
    ser.print(getStr); //envío del comando GET al dispositivo ESP8266
    Serial.println(getStr);  //Visualización del comando enviado en el monitor serie
    }
  else{
    ser.println("AT+CIPCLOSE"); //en caso de no poder enviar el comando, enviar al ESP8266 el cierre de la conexión TCP
    Serial.println("AT+CIPCLOSE"); //Alerta del error en el envío del comando por el monitor serie
  }
  //Espera en ms para volver a ejecutar el bucle completo y enviar la información de las 6 variables
  delay(60000);
}

//---------------- Funciones para el ADXL345
//Escribe val a la dirección de registro del dispositivo
void writeTo(int device, byte address, byte val) {
  Wire.beginTransmission(device); //iniciar transmisión al dispositivo ADXL345
  Wire.write(address);      //enviar dirección de registro al ADXL345
  Wire.write(val);      //enviar el valor a escribir al ADXL345
  Wire.endTransmission(); //finalizar transmisión al ADXL345
}

//leer el numero bytes iniciando desde la dirección de registro al dispositivo en el arreglo de buffer
void readFrom(int device, byte address, int num, byte buff[]) {
  Wire.beginTransmission(device); //iniciar transmisión al dispositivo ADXL345
  Wire.write(address);      //enviar dirección de registro al ADXL345
  Wire.endTransmission(); //finalizar transmisión al ADXL345

    Wire.beginTransmission(device); //iniciar transmision al dispositivo ADXL345
  Wire.requestFrom(device, num);  //solicitud de 6 bytes desde el dispositivo

  int i = 0;
  while(Wire.available()) //dispositivo quizá envié menos de lo requerido
  {
    buff[i] = Wire.read(); // recibe un byte
    i++;
```

```
  }
  Wire.endTransmission(); //finaliza la transmisión
}

//cálculo del  Roll y el Pitch
void RP_calculate(){
  double x_Buff = float(x);
  double y_Buff = float(y);
  double z_Buff = float(z);
  roll = atan2(y_Buff , z_Buff) * 57.3;
  pitch = atan2((- x_Buff) , sqrt(y_Buff * y_Buff + z_Buff * z_Buff)) * 57.3;
}
```

Anexo G. AT comandos de ejemplo (esp8266 at command examples).

ESP8266 AT Command Examples

Version 1.3

Espressif Systems IOT Team

Copyright © 2015

Disclaimer and Copyright Notice

Information in this document, including URL references, is subject to change without notice.

THIS DOCUMENT IS PROVIDED AS IS WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. All liability, including liability for infringement of any proprietary rights, relating to use of information in this document is disclaimed. No licenses express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

The WiFi Alliance Member Logo is a trademark of the WiFi Alliance.

All trade names, trademarks and registered trademarks mentioned in this document are property of their respective owners, and are hereby acknowledged.

Copyright © 2015 Espressif Systems Inc. All rights reserved.

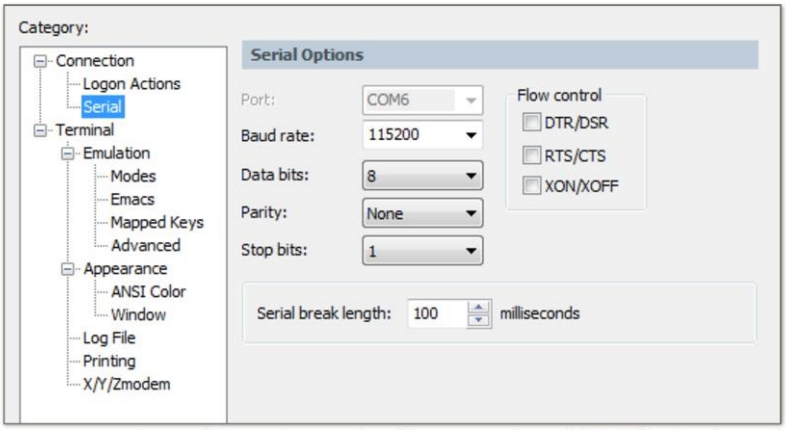
Table of Contents

1. Preambles

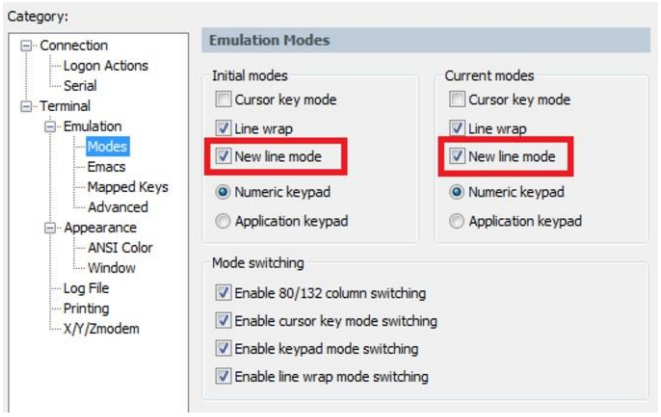
Herein we introduces some specific examples on the usage of Espressif AT Commands. For more information about the complete instruction set, please refer to documentation " 4A-ESP8266__AT Instruction Set ".

(1) Flash in AT bin files which supports AT commands (`/esp_iot_sdk/bin/at`) into the ESP8266 device, according to "readme.txt" there.

(2) Power on device and set serial baud rate to 115200. Enter AT commands.



Note: Please pay attention to the new line mode, AT command need "`/r/n`" to be the end.



2. Single Connection as TCP Client

- **Set WiFi mode**:

```
AT+CWMODE=3      // softAP+station mode
Response :OK
```

- **Connect to router**:

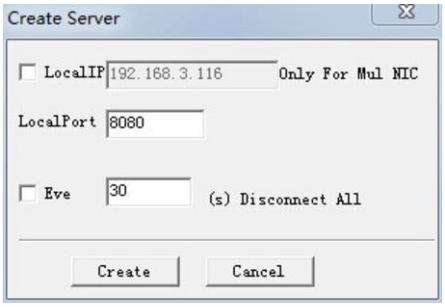
```
AT+CWJAP="SSID", "password"       // SSID and password of router
Response :OK
```

- **Query device's IP**:

```
AT+CIFSR
Response :192.168.3.106   // Device got an IP from router.
```

- Connect PC to the same router that ESP8266 is connected to.

  Using a network tool on the computer to create a server.



- **ESP8266 connect to server as a client**:

```
AT+CIPSTART="TCP", "192.168.3.116", 8080  //protocol, server IP & port
Response :OK
```

- **Send data**:

```
AT+CIPSEND=4   // set date length which will be sent,  such as 4 bytes

>DGFY         // enter the data,  no CR
Response :SEND OK
```

Note: If the number of bytes sent is bigger than the size defined (n), will reply busy, and after sending n number of bytes, reply SEND OK.

- **Receive data**:

```
+IPD, n: xxxxxxxxxx    // received n bytes,  data=xxxxxxxxxx
```

3. UDP Transmission

UDP transmission is established via AT+CIPSTART. There is no such definition as UDP server or UDP client. For more information about AT commands, please refer to documentation " 4A-ESP8266__AT Instruction Set ".

- **Set WiFi mode** :

```
AT+CWMODE=3      // softAP+station mode
Response :OK
```

- **Connect to router**:

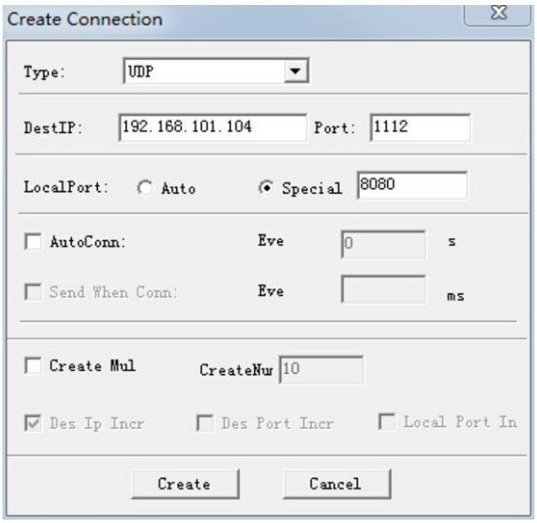
```
AT+CWJAP="SSID", "password"       // SSID and password of router
Response :OK
```

- **Query device's IP**:

```
AT+CIFSR
Response :+CIFSR: STAIP, "192.168.101.104" // IP address of ESP8266 station
```

- Connect PC to the same router as ESP8266 is connected to.

  Using a network tool on the computer to create a UDP .



Below is two examples on UDP transmission.

3.1. UDP (remote IP and port are fixed)

In UDP transmission, whether remote IP and port is fixed or not is decided by the last parameter of "AT+CIPSTART". "0" means that the remote IP and port is fixed and cannot be changed. A specific ID is given to such connection, making sure that the data sender and receiver will not be replaced by other devices.

- **Enable multiple connection**:

```
AT+CIPMUX=1
Response :OK
```

- **Create a UDP transmission, for example, ID is 4.**

```
AT+CIPSTART=4, "UDP", "192.168.101.110", 8080, 1112, 0
Response :4, CONNECT   OK
```

Note :

"192.168.101.110", 8080 here is the remote IP and port of UDP transmission of the opposite side, i.e., the configuration set by PC.

1112 is the local port of ESP8266. User can self-define this port. The value of this port will be random if it's not defined beforehand.

0 means that remote IP and port is fixed and cannot be changed. For example, if another PC also creates a UDP entity and sends data to ESP8266 port 1112. ESP8266 can receive data sent from UDP port 1112, but when data is sent using AT command "AT+CIPSEND=4, X", it will still be sent to the first PC end. If this parameter is not 0, it will send to a new PC.

- **Send data**:

```
AT+CIPSEND=4, 5    // Send 5 bytes to transmission NO.4
>DGFYQ          // enter the data,  no CR
Response :SEND OK
```

Note:

If the number of bytes sent is bigger than the size defined (n), will reply busy, and after sending n number of bytes, reply SEND OK.

- **Receive data**:

```
+IPD, 4, n: xxxxxxxxxx     // received n bytes,  data=xxxxxxxxxxx
```

- **Delete UDP transmission NO.4**:

```
AT+CIPCLOSE=4
Response :4, CLOSED   OK
```

3.2. UDP (remote IP, port can be changed)

- **Create a UDP transmission, last parameter is "2".**

```
AT+CIPSTART="UDP", "192.168.101.110", 8080, 1112, 2
Response :CONNECT   OK
```

Note :

"192.168.101.110", 8080 here refer to the remote IP and port of UDP transmission terminal which is created on PC in step 4;

1112 is the local port of ESP8266. User can self-define this port. The value of this port will be random if it's not defined beforehand.

2 means the opposite terminal of UDP transmission side will change to be the latest one that has been communicating with ESP8266.

- **Send data**:

```
AT+CIPSEND=5   // Send 5 bytes

>DGFYQ         // enter the data,  no CR
Response :SEND OK
```

Note:

If the number of bytes sent is bigger than the size defined (n), will reply busy, and after sending n number of bytes, reply SEND OK.

- If you want to send data to any other UDP terminals, please set the IP and port of this terminal.

```
AT+CIPSEND=6, "192.168.101.111", 1000   // Send 6 bytes

>abcdef         // enter the data,  no CR
Response :SEND OK
```

- **Receive data**:

```
+IPD, n: xxxxxxxxxx    // received n bytes,  data=xxxxxxxxxx
```

- **Delete UDP transmission**:

```
AT+CIPCLOSE
Response :CLOSED   OK
```

4. Transparent Transmission

Transparent transmission is enabled only when ESP8266 is working as TCP client creating a single connection, or in UDP transmission.

4.1. TCP client single connection UART - WiFi passthrough

Here is an example that ESP8266 station as TCP client to create a single connection and execute transparent transmission. For ESP8266 soft-AP, it can execute transparent transmission in the similar way. For more information about AT commands, please refer to documentation " 4A-ESP8266__AT Instruction Set ".

- **Set WiFi mode** :

```
AT+CWMODE=3      // softAP+station mode
Response :OK
```

- **Connect to router**:

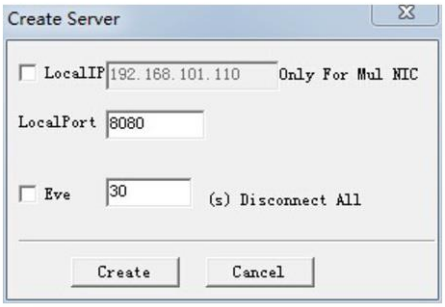
```
AT+CWJAP="SSID", "password"       // SSID and password of router
Response :OK
```

- **Query device's IP**:

```
AT+CIFSR
Response :192.168.101.105   // Device's IP that got from router.
```

- Connect PC to the same router that ESP8266 is connected to.

  Using a network tool on the computer to create a server.



- **Device connect to server**:

```
AT+CIPSTART="TCP", "192.168.101.110", 8080  // protocol、server IP & port
Response :OK     Linked
```

- **Enable transparent transmission mode**:

```
AT+CIPMODE=1
Response :OK
```

- **Start sending data**:

```
AT+CIPSEND
    Response:  >          //From now on,  data received from UART will be
transparent transmitted to server.
```

The network tool on PC will receive the data.

- **Stop sending data**:

If received a packet of data that contains only "+++", then the transparent transmission process will be stopped. Please wait at least 1 second before sending next AT command.

Please be noted that if you input "+++" directly by typing, the "+++", may not be recognised as three consecutive "+" because of the Prolonged time when typing.

Note:

The aim of ending "+++" is to exit transparent transmission and turn back to accept normal AT command, while TCP still remains connected. However, we can also use command "AT+CIPSEND" to turn back into transparent transmission.

- **Disable UART - WiFi passthrough mode (transparent transmission)**

```
AT+CIPMODE=0
Response :  OK
```

- **Delete TCP connection**:

```
AT+CIPCLOSE
Response :CLOSED   OK
```

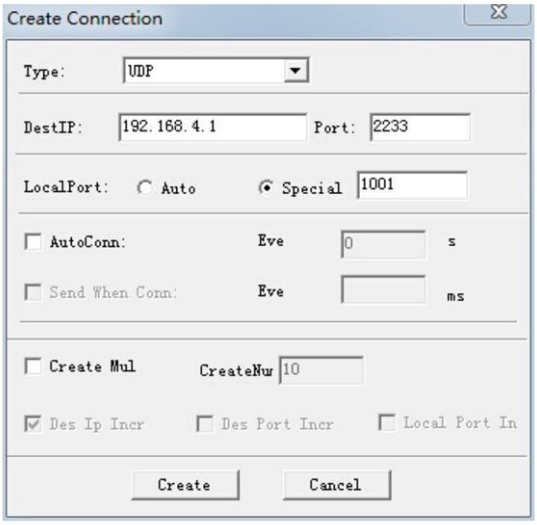
4.2. UDP transmission UART - WiFi passthrough

Here is an example that ESP8266 soft-AP create a UDP transparent transmission. For ESP8266 station, it can execute UDP transparent transmission in the similar way. For more information about AT commands, please refer to documentation " 4A-ESP8266__AT Instruction Set ".

- **Set WiFi mode** :

```
AT+CWMODE=3      // softAP+station mode
Response :OK
```

- **Connect PC to ESP8266 soft-AP**
- Using a network tool on PC to create a UDP.



- **Device create a UDP transmission which remote IP and port are fixed.**

```
AT+CIPSTART="UDP","192.168.4.2",1001,2233,0
Response :OK
```

- **Enable transparent transmission mode**:

```
AT+CIPMODE=1
Response :OK
```

- **Start sending data**:

```
AT+CIPSEND
```

```
        Response:  >          //From now on,  data received from UART will be
    transparent transmitted to server.
```

- **Stop sending data**:

If a packet of data that contains only "+++", then the transparent transmission process will be stopped. Please wait at least 1 second before sending next AT command.

Please be noted that if you input "+++" directly by typing, the "+++", may not be recognised as three consecutive "+" because of the Prolonged time when typing.

Note:

The aim of ending "+++" is to exit transparent transmission and turn back to accept normal AT command, while UDP transmission still exists. However, we can also use command "AT+CIPSEND" to turn back into transparent transmission.

- **Disable UART - WiFi passthrough mode (transparent transmission)**

```
AT+CIPMODE=0
Response :  OK
```

- **Delete UDP transmission**:

```
AT+CIPCLOSE
Response :CLOSED   OK
```

5. Multiple Connection as TCP Server

When ESP8266 is working as a TCP server, a multiple of connections shall be maintained. That is to say, there should be more than one client connecting to ESP8266.

Here is an example showing how TCP server is realized when ESP8266 is working in softAP mode:

- **Set WiFi mode** :

```
AT+CWMODE=3      // softAP+station mode
Response :OK
```

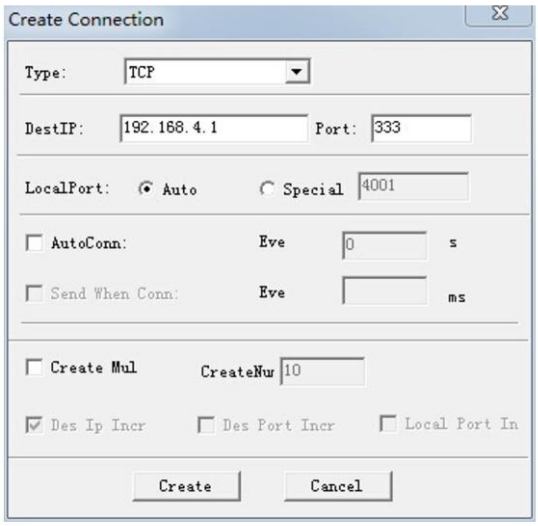
- **Enable multIPle connection**:

```
AT+CIPMUX=1
Response :OK
```

- **Setup server**:

```
AT+CIPSERVER=1       // default port = 333
Response :OK
```

- **Connect PC to ESP8266 soft-AP**
- Using a network tool on PC to create a TCP client and connect to ESP8266.



Note:

When ESP8266 is working as a TCP server, there exists a timeout mechanism. If the TCP client is connected to the ESP8266 TCP server, whereas there is no data transmission for a period of time, then the server will stop the connection with the client. To avoid such problems, please set up a data transmission circulation every 5 seconds.

- **Send data**:

```
// ID number of connection is defaulted to be 0.
 AT+CIPSEND=0, 4   // send 4 bytes to connection NO.0

 >iopd          // enter the data,  no CR
 Response :SEND OK
```

Note:

If the number of bytes sent is bigger than the size defined (n), will reply busy, and after sending n number of bytes, reply SEND OK.

- **Receive data**:

```
  +IPD, 0, n: xxxxxxxxxx    // received n bytes,  data = xxxxxxxxxx
```

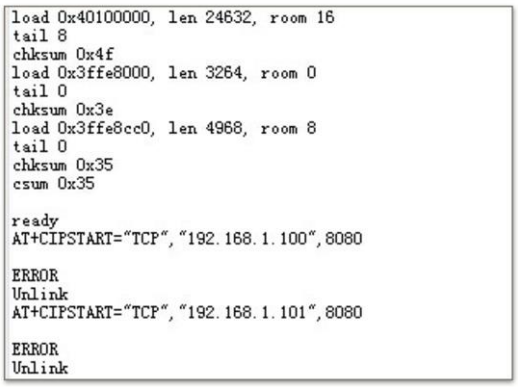
- **Delete one TCP connection**:

```
  AT+CIPCLOSE=0    // Delete NO.0 connection.
Response :0, CLOSED    OK
```

6. Questions & Answers

If you have any questions about the execution of AT instructions, please contact support-at@espressif.com. Please describe the issues that you encountered with information as follows:

- Version info or AT Command: You can use command "AT+GMR" to acquire your current AT command version info.
- Hardware Module info :example AITHINK ESP-01
- Screenshot of the test steps, for example:

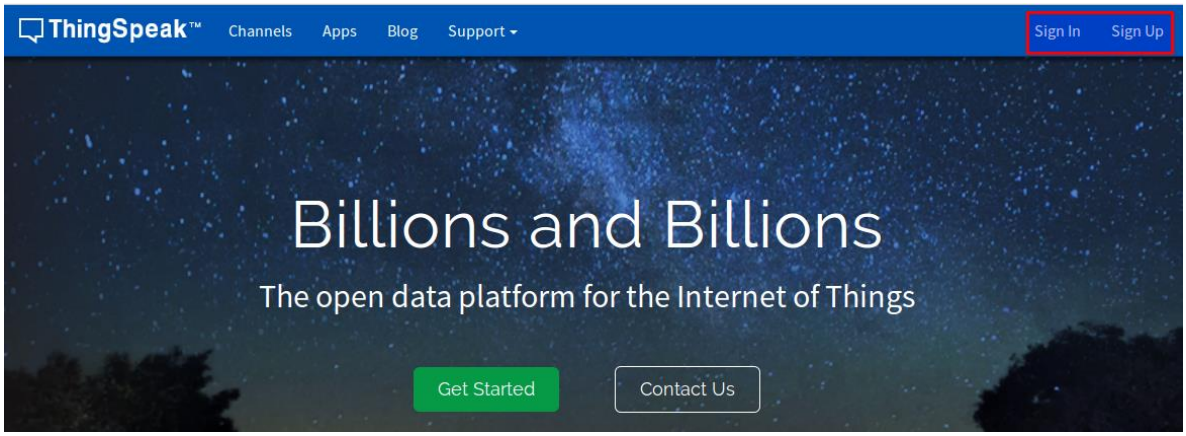


- If possible, please provide the printed log information, such as:

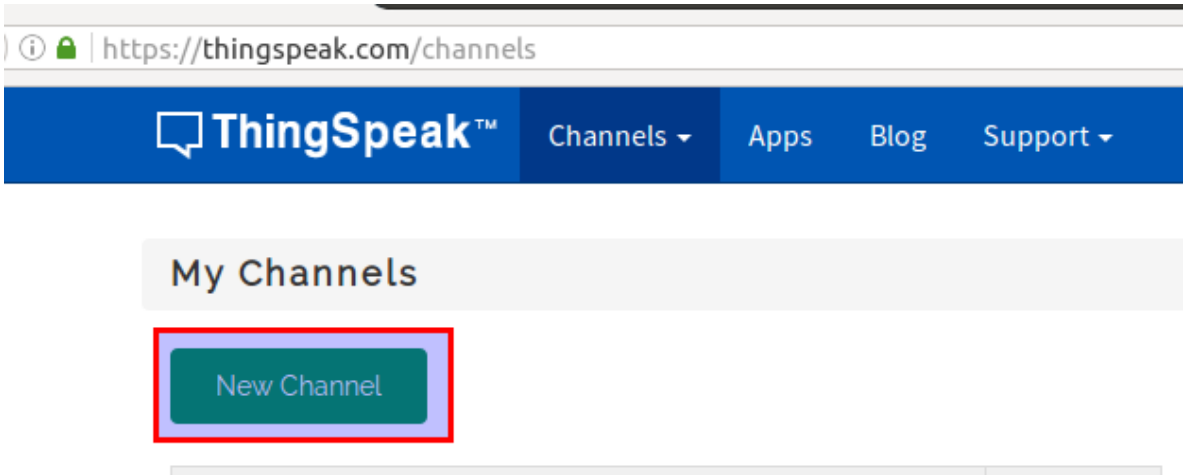
```
ets Jan  8 2013, rst cause: 1,  boot mode: (3, 3)
load 0x40100000,  len 26336,  room 16
tail 0
chksum 0xde
load 0x3ffe8000,  len 5672,  room 8
tail 0
chksum 0x69
load 0x3ffe9630,  len 8348,  room 8
tail 4
chksum 0xcb
csum 0xcb
SDK version: 0.9.1
addr not ack when tx write cmd
```

Anexo H. Pasos iniciales en ThingSpeak

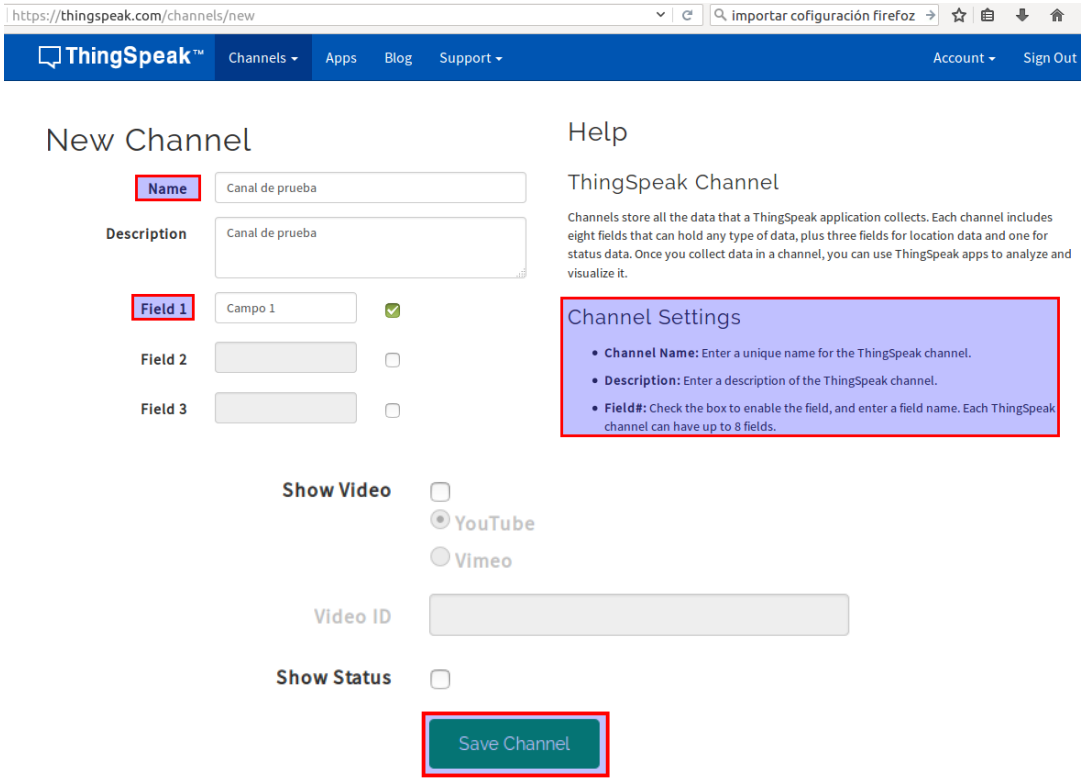
I. Ingresar a https://thingspeak.com/, crear una cuenta (*Sign Up*) o iniciar sesión (*Sign In*) en la plataform:



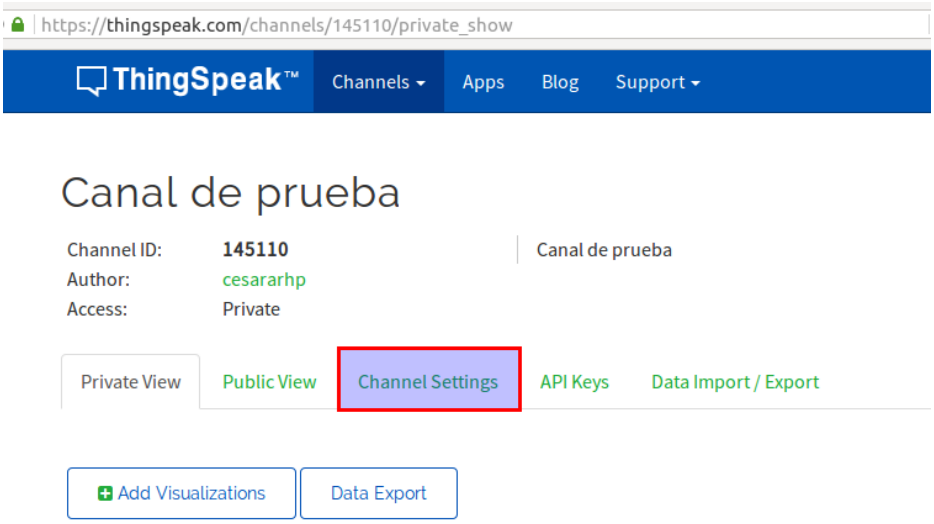
II. Posterior al ingreso, vamos a crear un canal dando clic en el botón “New cannel” de la página principal:

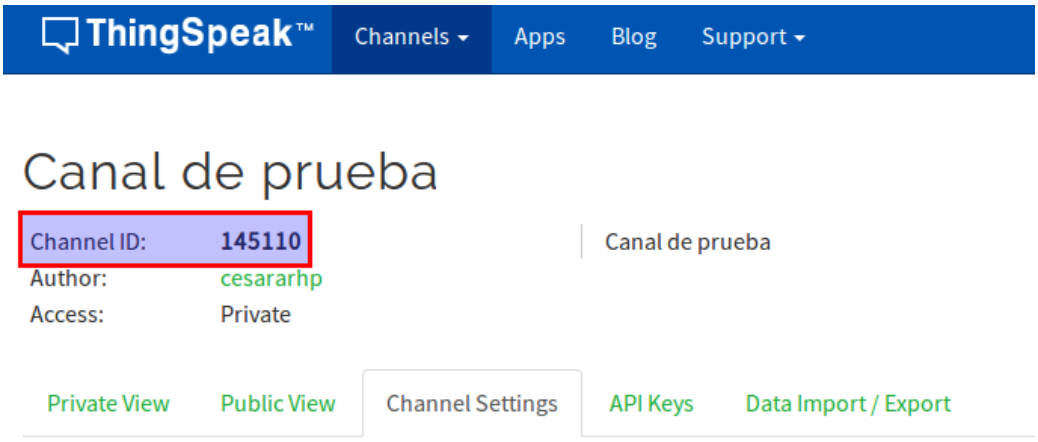


III. Diligenciamos el campo Nombre y seleccionamos como mínimo uno de los campos para la recepción de la información, al cual también le damos un nombre. El formulario posee 8 campos e información adicional, opcional para habilitar y modifica otras características del canal, las cuales poseen una breve ayuda/descripción en la parte derecha. Finalizamos dando clic en el botón “*Save Channel*” de la parte inferior de la página:

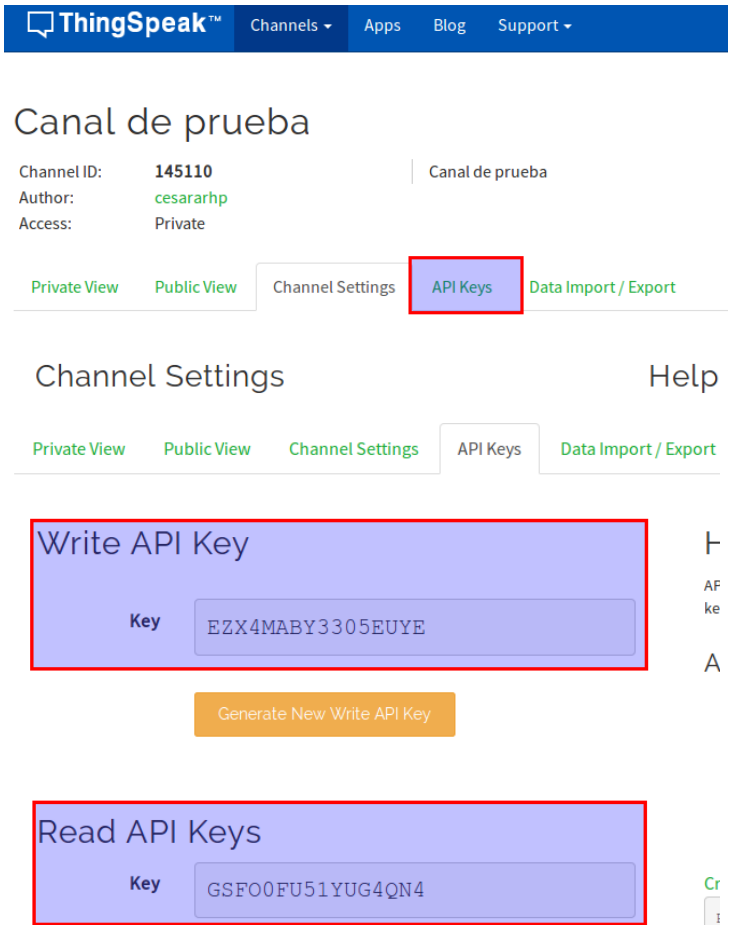


IV. Posterior a crear el canal, se nos muestra la información relevante del mismo, en la pestaña "*Channel Settings*", allí podremos consultar el ID del canal, habilitar más campos de ser necesario y modificar la información ingresada en el formulario original:



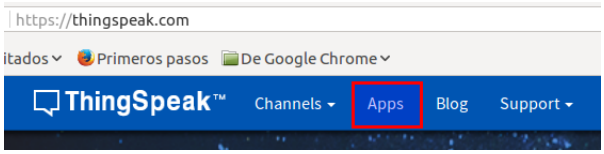


V. De igual forma, a través de la pestaña "*API Keys*" es posible consultar la información requerida para escribir o leer la información del canal:

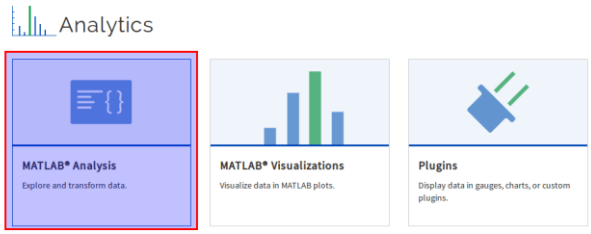


Anexo I. *MATLAB Analysis* y *TimeControl* para calcular las diferencias entre medidas.

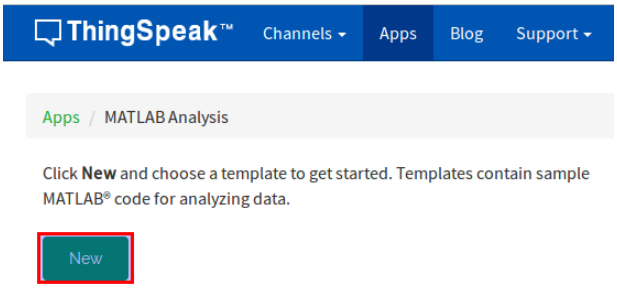
I. Iniciar sesión en ThingSpeak.com y en el menú de la parte superior, dar clic en Apps.



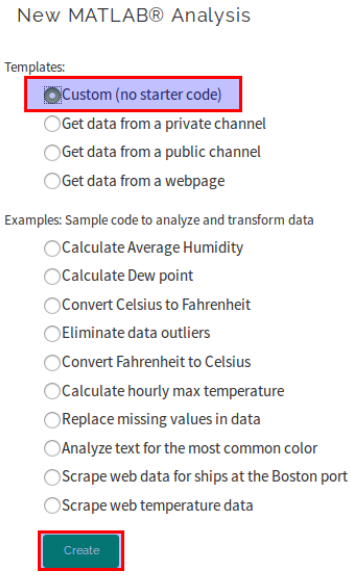
II. Del apartado *Analytics*, seleccionar *MATLAB Analysis*.



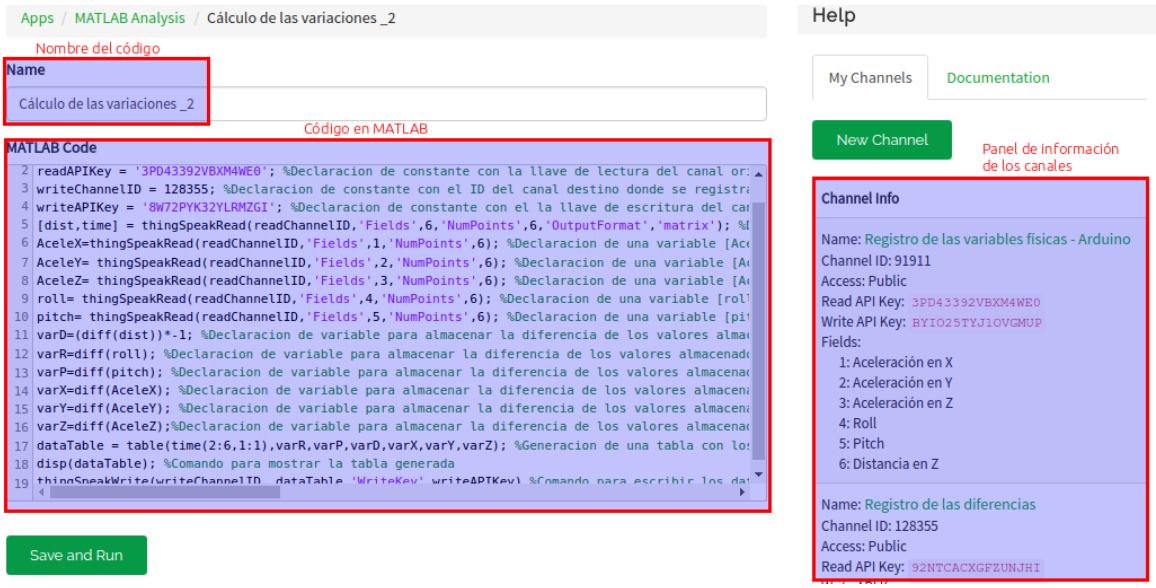
III. Dar clic en el botón New para crear un nuevo análisis de datos:



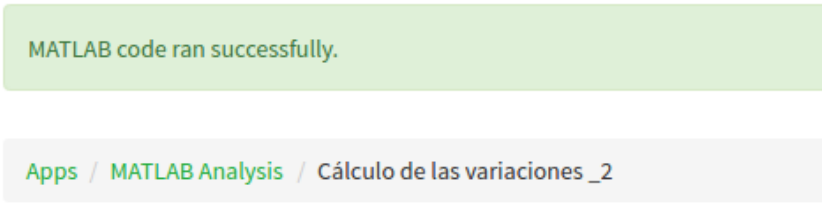
IV. Seleccionar la opción *Custom(no starter code)* y dar clic en el botón *Create*.



V. La página que se mostrará tendrá un campo de nombre para referenciar el código y un campo para digitar el código en MATLAB, para nuestro caso en particular, se nombró el código “Cálculo de las variaciones _2” y con el código para el cálculo de las diferencias ya mostrado previamente. En la parte derecha de la página, se muestra un panel con la información de los canales que se tienen creados para poder usarla dentro del código en MATLAB.

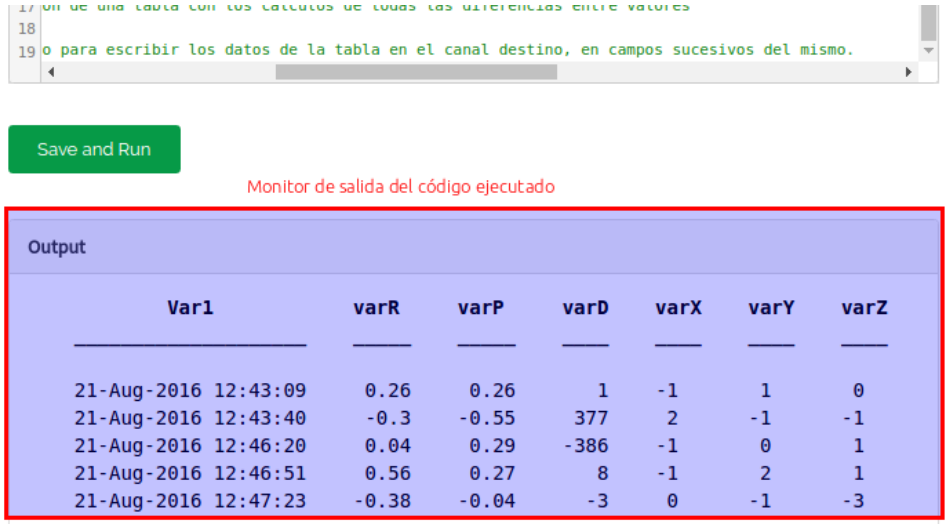


VI. Posterior al ingreso del código, deberemos dar clic en el botón Save and run de la parte inferior del campo del código, si el código no contiene errores y se ejecutó satisfactoriamente, en la parte superior de la página nos mostrará dicho mensaje.

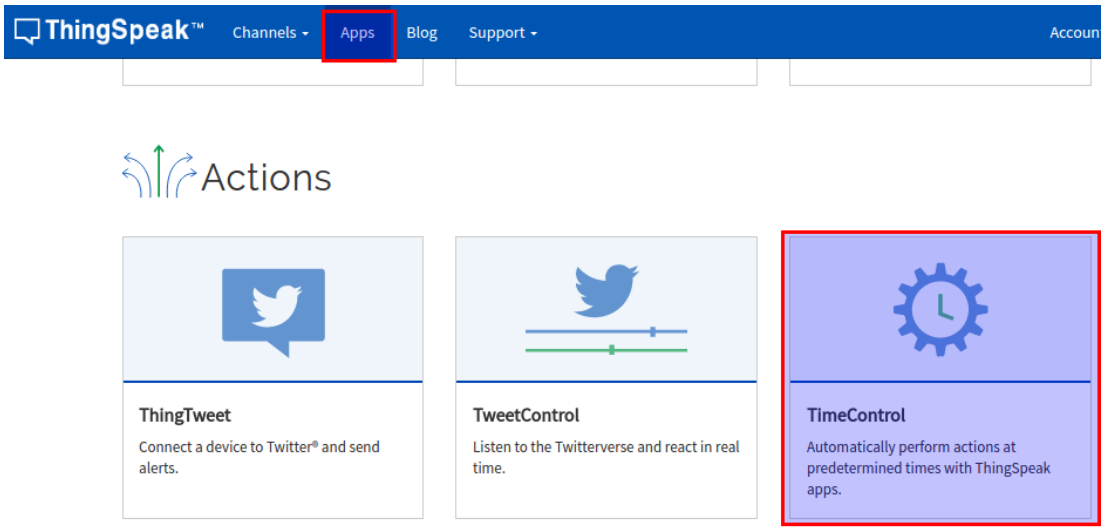


En este momento ya deberemos tener información almacenada en canal destino.

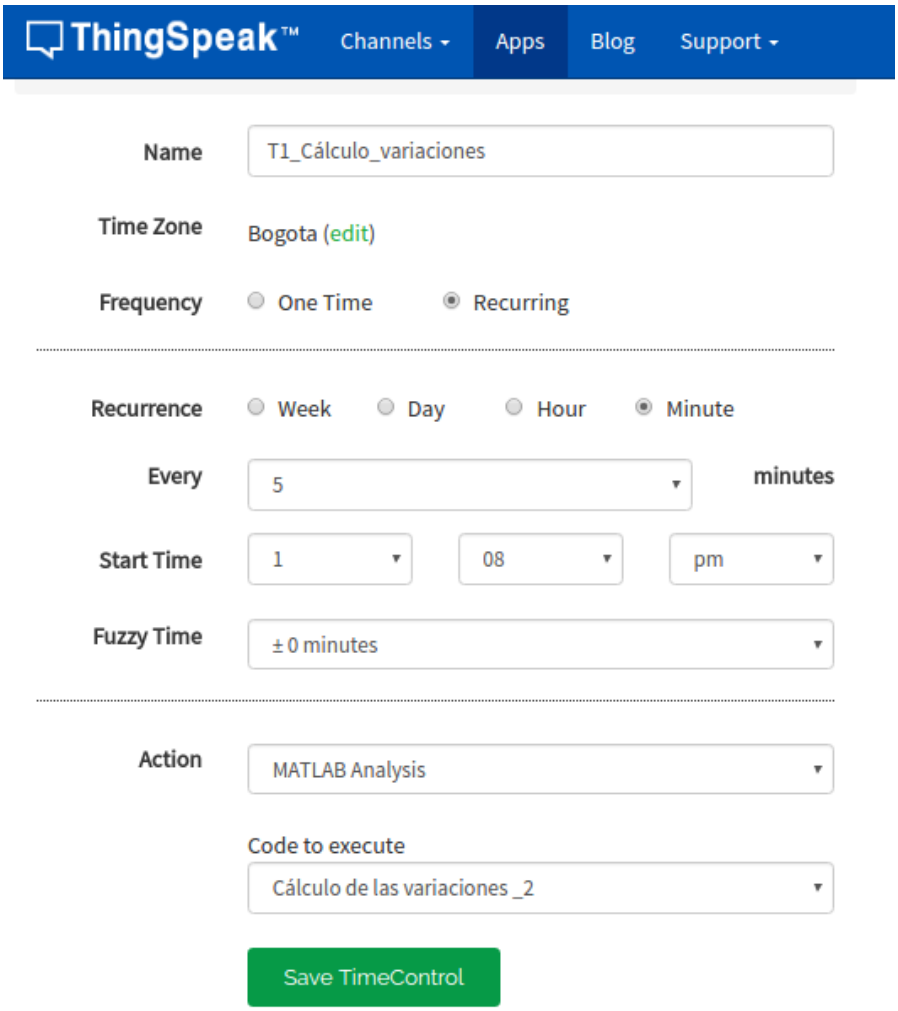
En caso dado de tener en el código comandos para mostrar información adicional o resultados, los mismos serán mostrados en el panel Output de la parte inferior, en nuestro caso el código tiene el comando disp para mostrar la tabla generada con las diferencias:



VII. Cómo se indicó en el apartado 6.2.3.3.1 es necesario programar la ejecución del código, para ello en la página principal de las Apps de ThingSpeak, en el apartado *Actions*, seleccionaremos *TimeControl*.

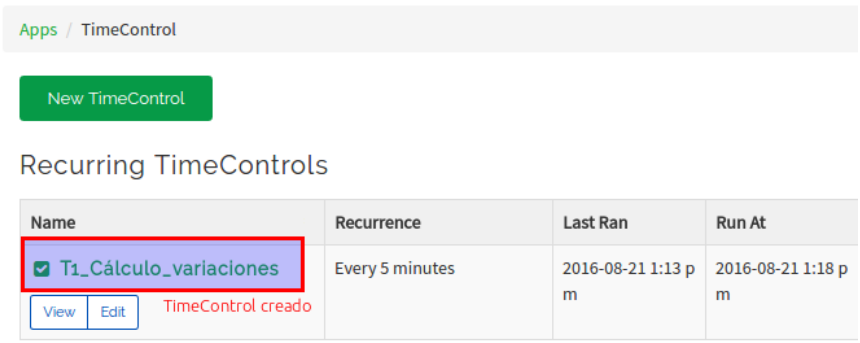


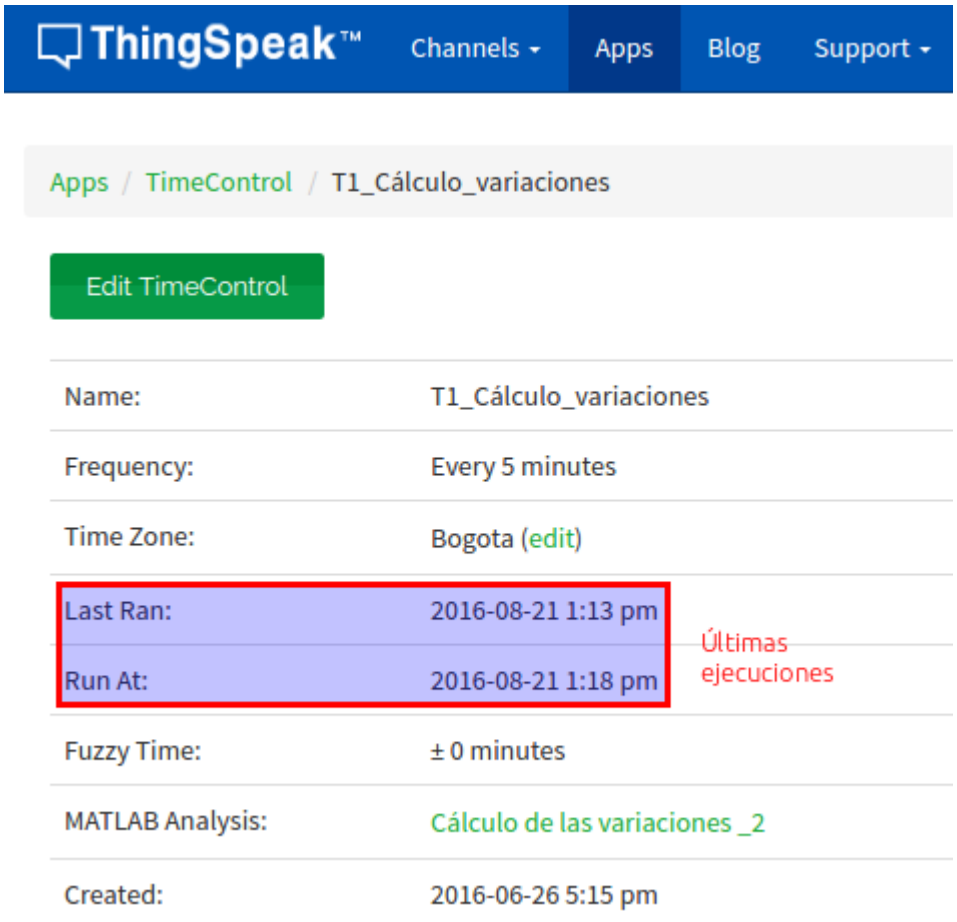
VIII. Damos clic en *New TimeControl* y programamos la ejecución automática del código previamente generado teniendo en cuenta que debemos asignarle un nombre al TimeControl, una zona horaria, indicar si será recurrente la ejecución o no, y la acción requerida, para nuestro caso el *TimeControl* queda de la siguiente forma:



Finalizando con el botón *Save TimeControl*.

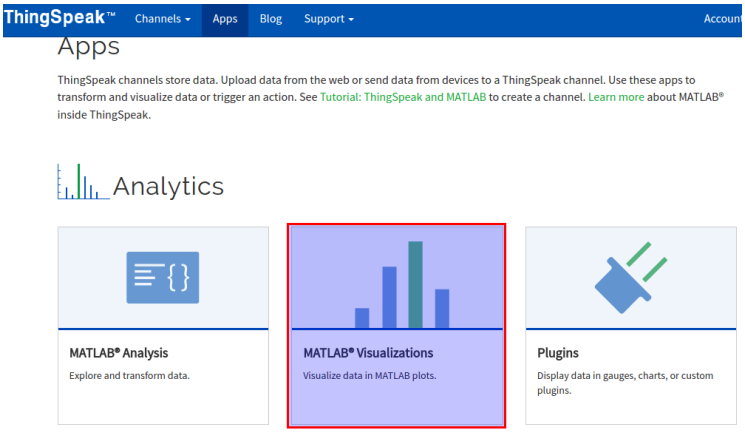
IX. El TimeControl creado quedará almacenado en la ventana principal, dando clic sobre el mismo nos mostrará información adicional de cómo está configurado y las fechas de ejecución:



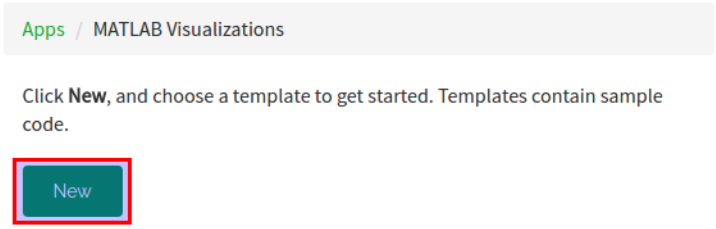

ThingSpeak™
Channels
Apps
Blog
Support
Apps / TimeControl / T1_Cálculo_variaciones
Edit TimeControl
Name: T1_Cálculo_variaciones
Frequency: Every 5 minutes
Time Zone: Bogota (edit)
Last Ran: 2016-08-21 1:13 pm
Run At: 2016-08-21 1:18 pm
Últimas ejecuciones
Fuzzy Time: ± 0 minutes
MATLAB Analysis: Cálculo de las variaciones _2
Created: 2016-06-26 5:15 pm

Anexo J. Uso e implementación de *MATLAB Visualizations*.

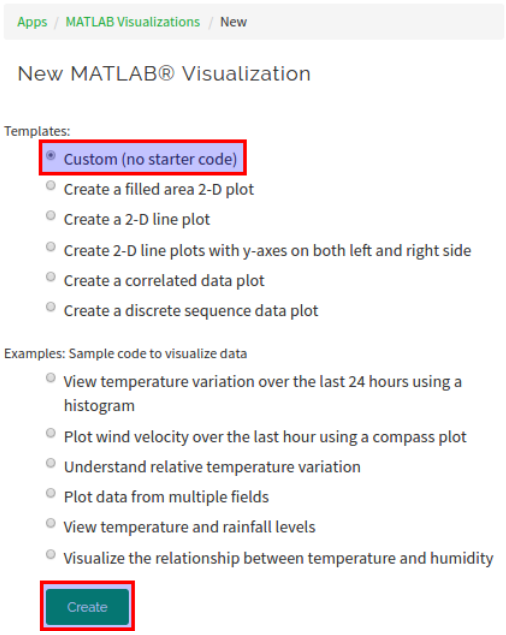
I. De la página principal de Apps, seleccionar *MATLAB Visualizations*.



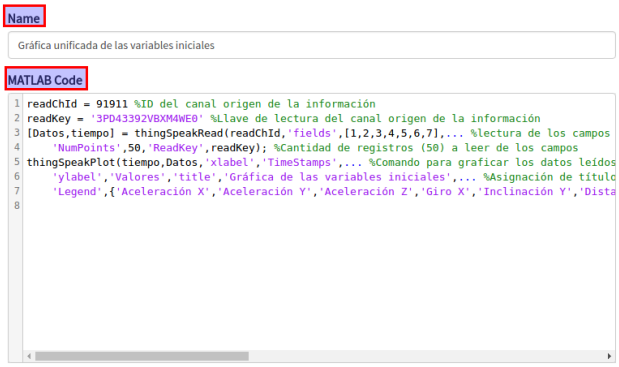
II. Dar clic en New para empezar a generar una nueva gráfica.



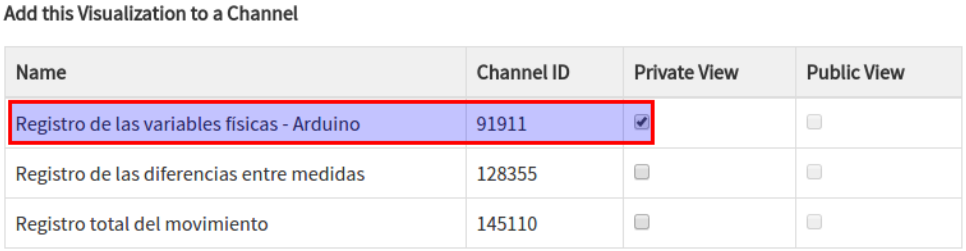
III. Seleccionar la opción *Custom (no starter code)* y dar clic en *Create*.



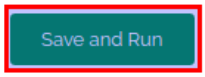
IV. Asignar un nombre al código e ingresar el código modelado para la generación de la gráfica.

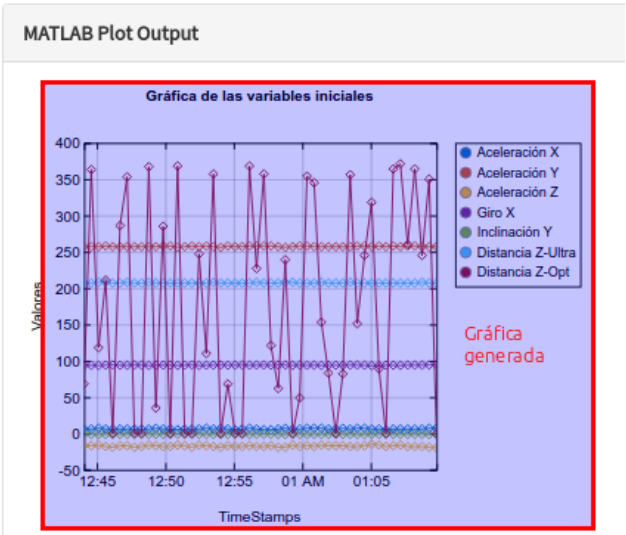


V. Seleccionaremos el canal sobre el cual adicionaremos la gráfica generada.

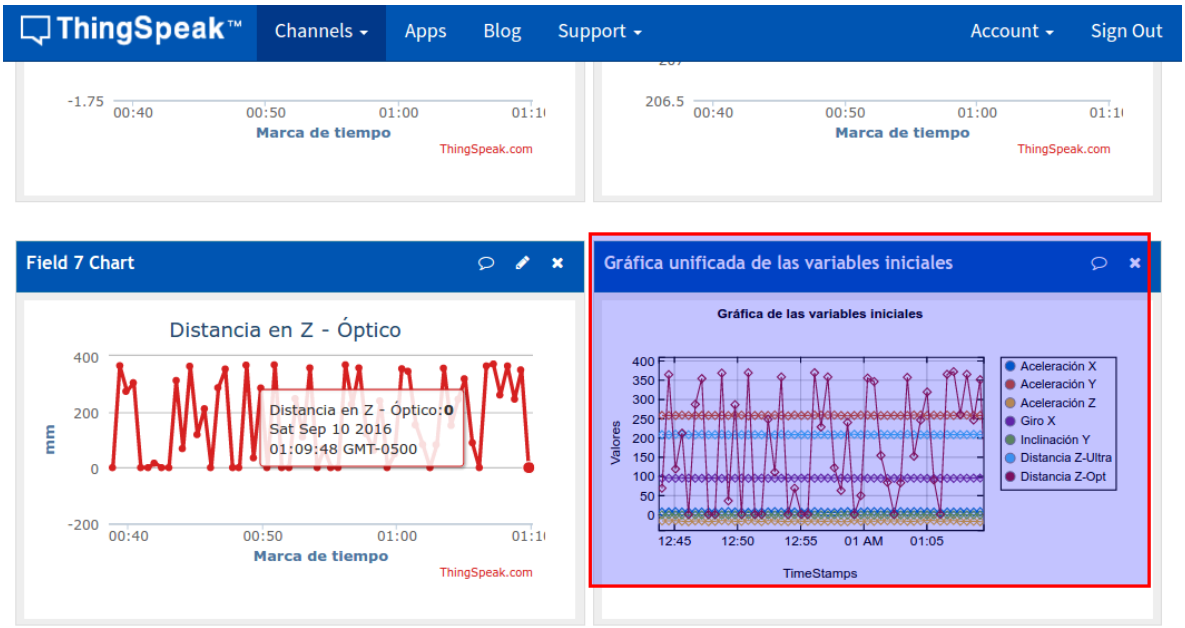


VI. Finalizaremos dando clic en el botón Save and Run, si el código es correcto en la parte inferior (*MATLAB Plot Output*) nos mostrará la gráfica generada.



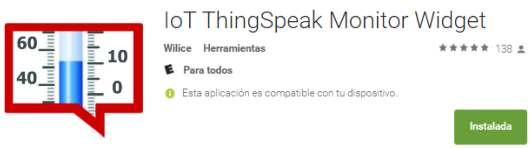


VII. De igual forma, si todo funcionó correctamente en el canal seleccionado en el paso V se deberá agregar la gráfica generada.

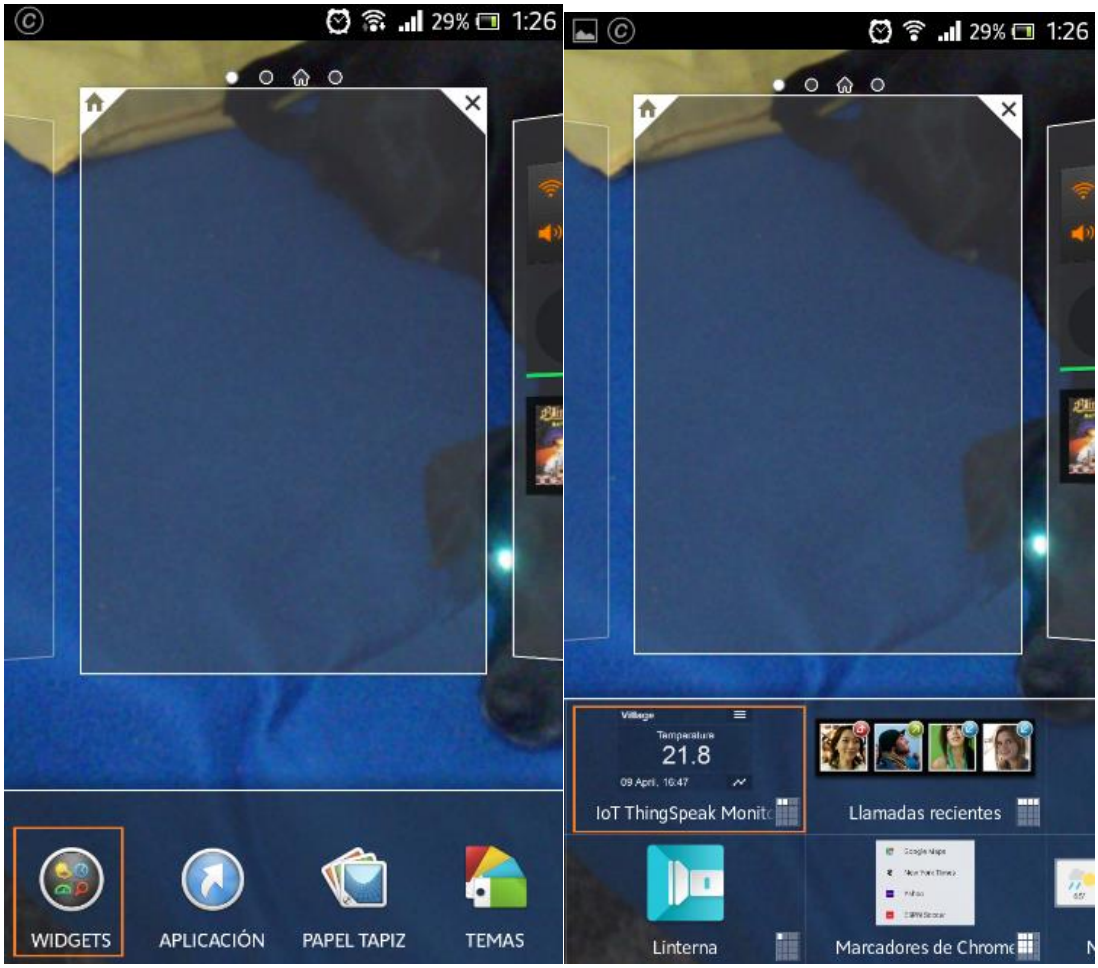

ThingSpeak™
Channels
Apps
Blog
Support
Account
Sign Out
-1.75
00:40
00:50
01:00
Marca de tiempo
ThingSpeak.com
206.5
00:40
00:50
01:00
Marca de tiempo
ThingSpeak.com
Field 7 Chart
Distancia en Z - Óptico
400
200
0
-200
mm
Distancia en Z - Óptico:0
Sat Sep 10 2016
01:09:48 GMT-0500
00:40
00:50
01:00
Marca de tiempo
ThingSpeak.com
Gráfica unificada de las variables iniciales
Gráfica de las variables iniciales
Valores
400
350
300
250
200
150
100
50
0
12:45
12:50
12:55
01 AM
01:05
TimeStamps
Aceleración X
Aceleración Y
Aceleración Z
Giro X
Inclinación Y
Distancia Z-Ultra
Distancia Z-Opt

Anexo K. Pruebas de IoT ThingSpeak Monitor Widget.

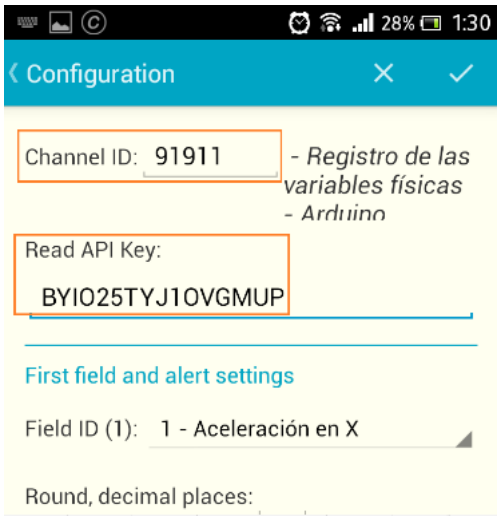
I. A través de Google Play Store realizar la búsqueda e instalación de la aplicación.



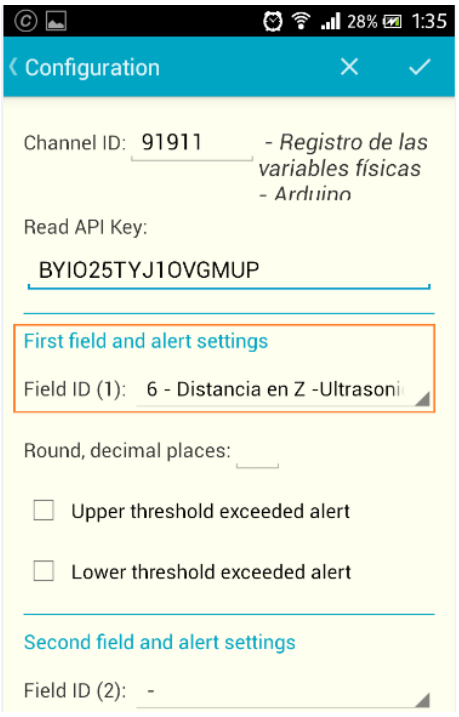
II. En la pantalla principal del dispositivo agregar el Widget.



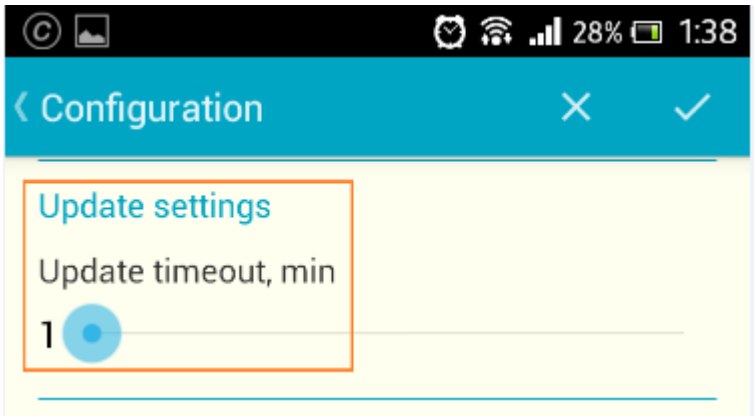
III. Al seleccionarlo, se desplegará la información básica de conexión al canal, en la cual deberemos suministrar el ID del canal (Channel ID), la llave de lectura (Read API Key), dicha información la debemos consultar del canal en la página de ThingSpeak.



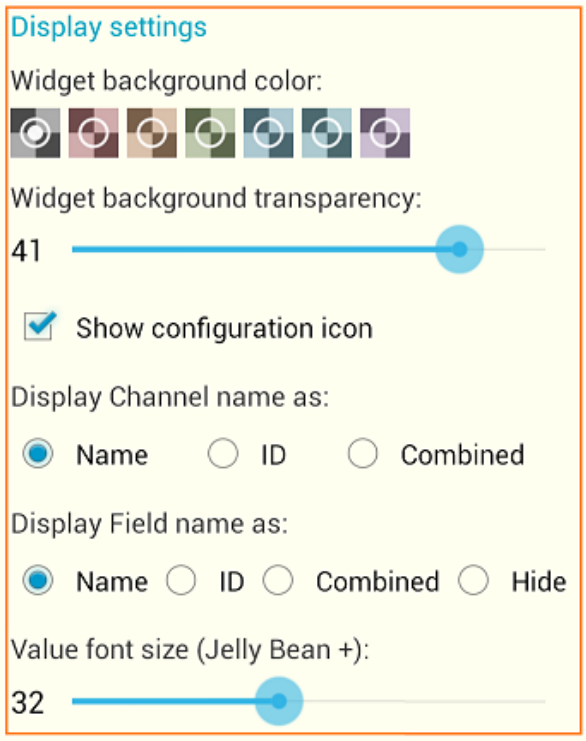
IV. La aplicación consultará el canal para traer los campos disponibles del mismo, nos permitirá agregar 2 campos por Widget generado. Para nuestro caso solamente seleccionaremos el valor del campo 6 para el Field ID (1).



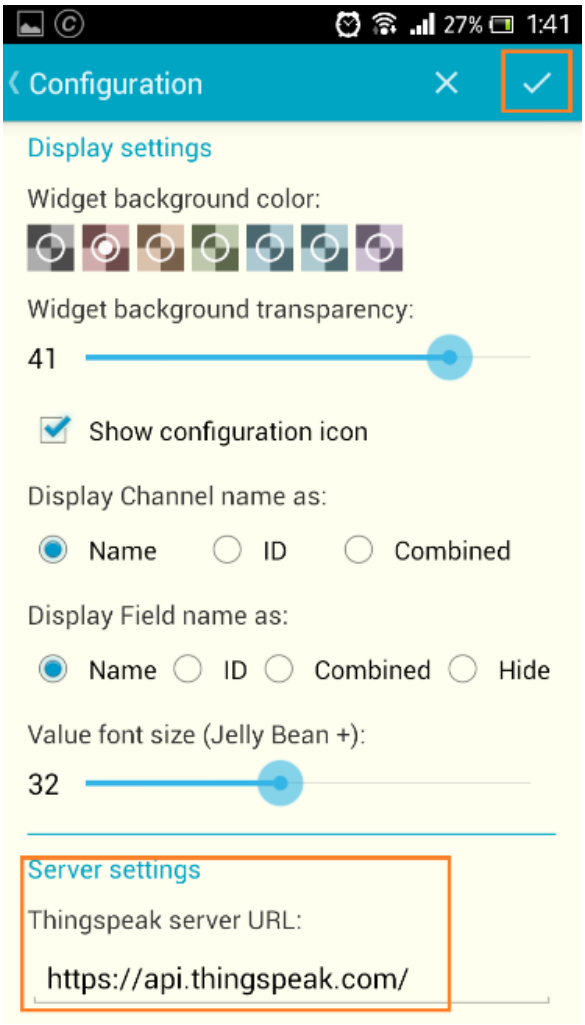
V. Nos desplazamos hasta el apartado *Update settings* en dónde cambiaremos la frecuencia en minutos con la que el Widget consultará información del canal para traer el valor del campo seleccionado. En nuestro la cambiaremos por 1 minuto.



VI. Los demás apartados de la aplicación cambian la apariencia con la que se mostrará el Widget en la pantalla.

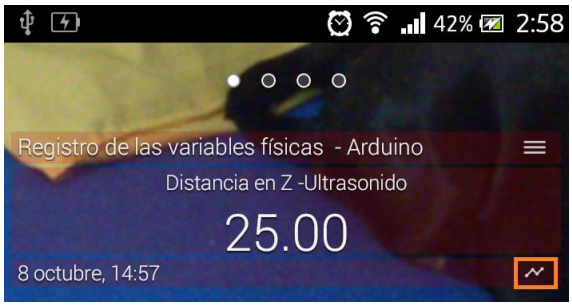


VII. En al apartado *Server settings* dejaremos el valor por defecto y finalizamos oprimiendo el ✓ de la parte superior derecha.



VIII. El widget aparecerá en pantalla, consultará la información y la veremos en pantalla.

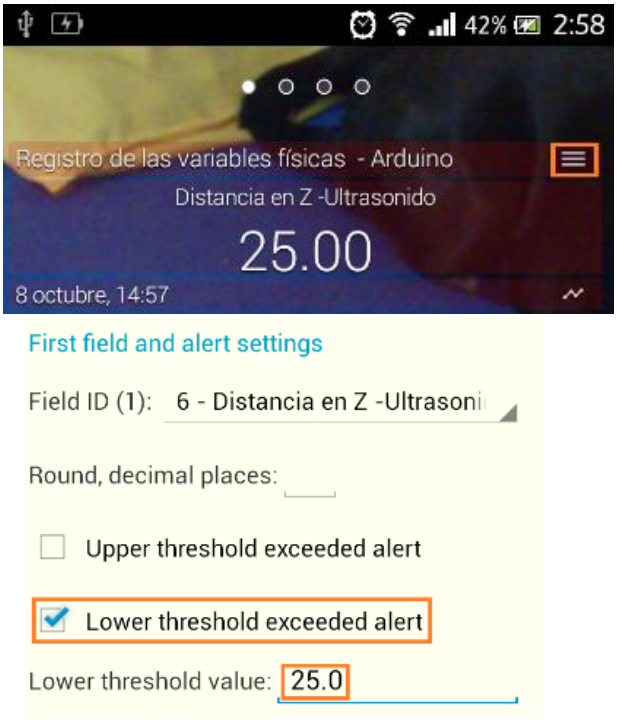
IX. El widget tiene la opción de mostrar la gráfica de los datos almacenados, para ello es necesario seleccionar la opción de la parte inferior derecha del widget.



X. El widget mostrará la gráfica del campo en particular que estamos consultando y posee la opción de promediarla en varias opciones de tiempo.

XI. Para la configuración de las alertas, entraremos de nuevo en la configuración del Widget, para lo cual seleccionaremos la opción de la parte superior derecha del mismo. Seleccionaremos los umbrales teniendo en cuenta si queremos indicar un valor por encima de (*Upper threshold exceded alert*) y menor que (*Lower threshold exceded alert)*, para nuestro ejemplo seleccionaremos el valor menor que y

estableceremos 25, para que nos alerte en caso dado de que tengamos un valor menor que el especificado. Guardamos cambios con el ✓ de la parte superior derecha.



XII. Al configurar la alerta, el Widget mostrará con una figura de campana, que dicho campo tiene una alerta configurada.

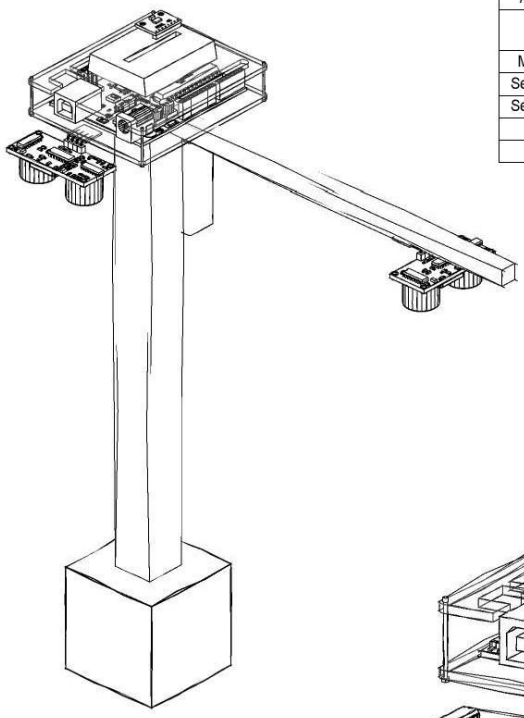
XIII. Forzamos una lectura menor que el valor indicado, y el Widget realizará una alerta sonora y un aviso en la bandeja del sistema indicándonos que el valor estuvo por debajo del establecido previamente.

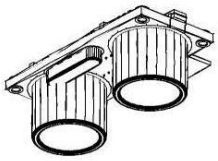
XIV. Cabe resaltar que es posible agregar dos campos por Widget generado, así cómo más de un Widget en la pantalla.

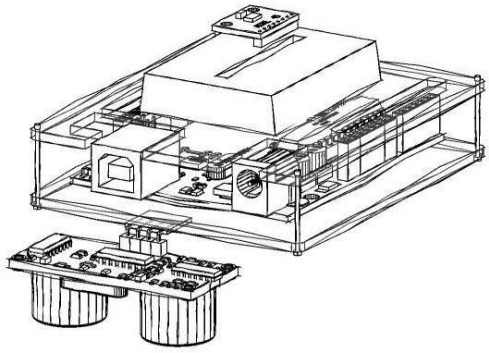
Anexo L. Planos del diseño de la estructura y prototipo.

Componentes del prototipo

Tipo	Modelo	Marca	Cantidad
Arduino Uno R3	Uno R3	Arduino	1
Acelerómetro ADXL345	ADXL 345	Analog Device	1
Mini BreadBoard	Genérico	Genérico	1
Sensor Ultrasónico	HCSR04	Genérico	1
Sensor Ultrasónico	HCSR04	Genérico	1
Batería	6LR61T 9V		1
Cambro			1







Lista de planos

Plano	Nombre de plano	Fecha	Versión
P1	Presentación	16/10/2016	4
P2	Planos prototipo y detalles	16/10/2016	4
P3	Plantas y cortes modelo escala	16/10/2016	4

Universidad Piloto de Colombia
UN ESPACIO PARA LA EVOLUCIÓN
Ingeniería de Telecomunicaciones

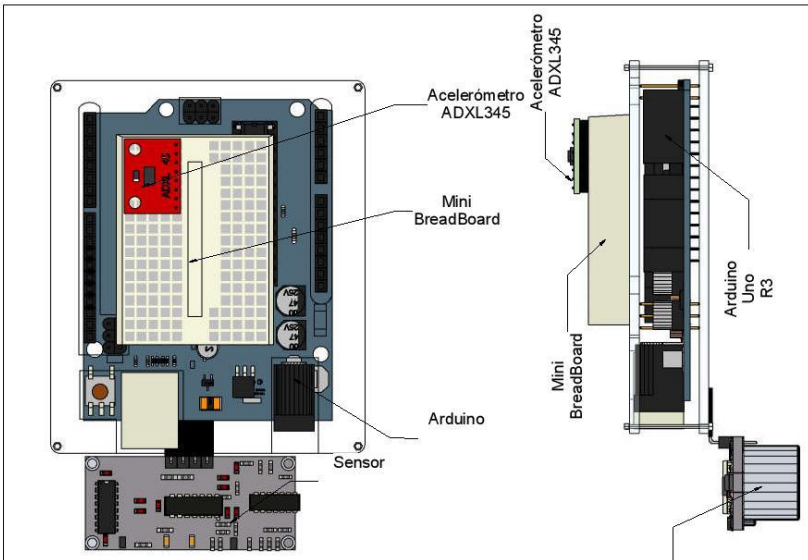
Prototipo para el monitoreo de estructuras en sistema de pórticos

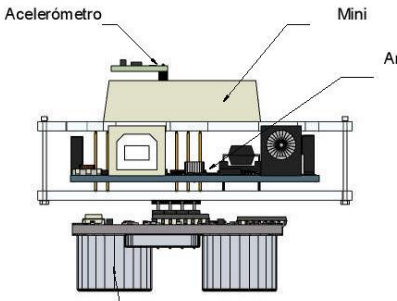
Versión	Descripción	Fecha
1	Emisión inicial	24/08/2016
2	Revisión general	29/08/2016
4	Correccion Detalles	16/10/2016

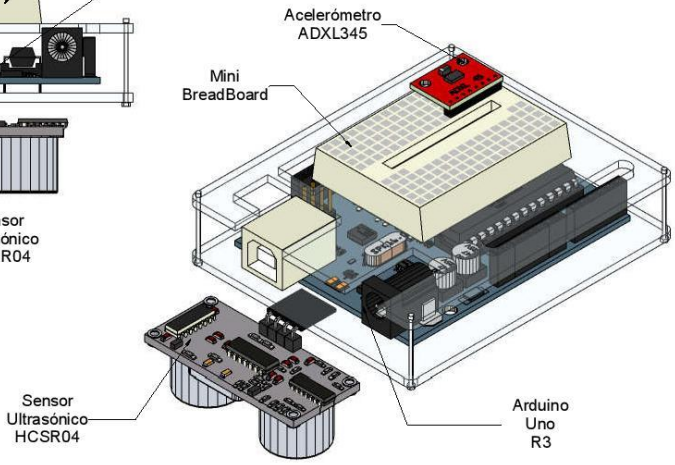
Presentación

Número de proyecto	0001	P1
Fecha	26/08/2016	
Dibujado por	CAHP	
Comprobado por	CAHP	Escala

16/10/2016 09:53:05 p.m.







Universidad Piloto de Colombia
UN ESPACIO PARA LA EVOLUCIÓN
Ingeniería de Telecomunicaciones

Prototipo para el monitoreo de estructuras en sistema de pórticos

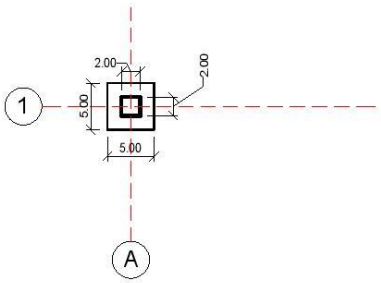
Versión	Descripción	Fecha
1	Emisión inicial	24/08/2016
2	Revisión general	29/08/2016
3	Revisión ejes y cotas	09/09/2016
4	Correccion Detalles	16/10/2016

Planos prototipo y detalles

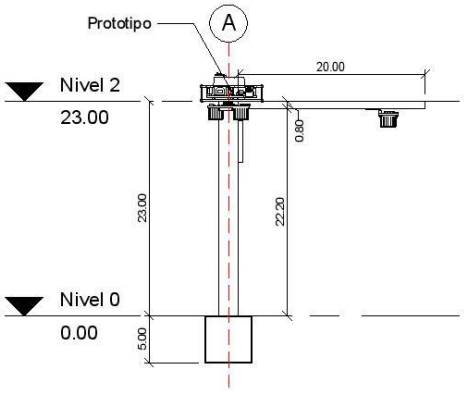
Número de proyecto	0001
Fecha	26/08/2016
Dibujado por	CAHP
Comprobado por	CAHP
Escala	1 : 1

P2

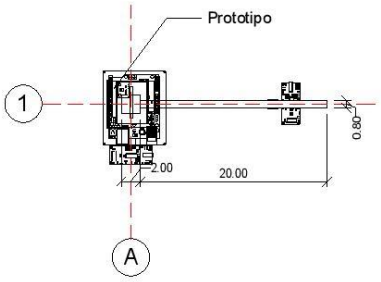
16/10/2016 09:53:09 p.m.



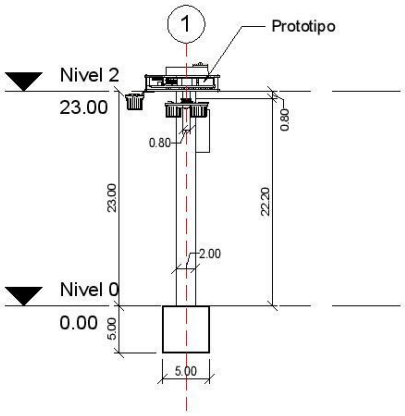
Nivel 0
Esc: 1 : 5



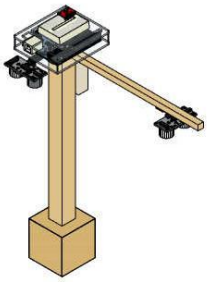
Vista 1
Esc: 1 : 5



Nivel 2
Esc: 1 : 5



Vista 2
Esc: 1 : 5



Modelo 3D
Esc:

Notas:
1. Medidas en centímetros (cm)
2. Detalles del sensor ver en plano P2
3. El modelo pertenece a una estructura real a escala 1:100.

Universidad Piloto de Colombia
UN ESPACIO PARA LA EVOLUCIÓN
Ingeniería de Telecomunicaciones

Prototipo para el monitoreo de estructuras en sistema de pórticos

Versión	Descripción	Fecha
1	Emisión inicial	24/08/2016
2	Revisión general	29/08/2016
3	Revisión ejes y cotas	09/09/2016
4	Correccion Detalles	16/10/2016

Plantas y cortes modelo escala

Número de proyecto	0001
Fecha	26/08/2016
Dibujado por	CAHP
Comprobado por	CAHP

P3

Escala 1 : 5

16/10/2016 09:53:14 p.m.

Anexo M: Medidas tomadas ultrasonidos

Hora	Distancia medida	% error
07:23:09 p.m.	27	0%
07:24:10 p.m.	27	0%
07:25:12 p.m.	27	0%
07:26:14 p.m.	27	0%
07:27:16 p.m.	27	0%
07:28:17 p.m.	27	0%
07:29:19 p.m.	27	0%
07:30:22 p.m.	27	0%
07:31:23 p.m.	27	0%
07:32:25 p.m.	28	4%
07:33:27 p.m.	27	0%
07:34:28 p.m.	28	4%
07:35:30 p.m.	28	4%
07:36:32 p.m.	28	4%
07:37:33 p.m.	28	4%
07:38:35 p.m.	27	0%
07:39:38 p.m.	27	0%
07:40:39 p.m.	28	4%
07:41:41 p.m.	27	0%
07:42:43 p.m.	27	0%
07:43:45 p.m.	27	0%
07:44:46 p.m.	27	0%
07:45:48 p.m.	27	0%
07:46:50 p.m.	27	0%
07:47:52 p.m.	27	0%
07:48:53 p.m.	27	0%
07:49:55 p.m.	27	0%
07:50:57 p.m.	28	4%
07:51:58 p.m.	27	0%
07:53:00 p.m.	27	0%
07:54:02 p.m.	28	4%
07:55:03 p.m.	27	0%
07:56:05 p.m.	27	0%
07:57:07 p.m.	27	0%
07:58:09 p.m.	27	0%
07:59:10 p.m.	28	4%
08:00:12 p.m.	27	0%
08:01:14 p.m.	27	0%
08:02:15 p.m.	28	4%
08:03:17 p.m.	27	0%
08:04:19 p.m.	28	4%
08:05:21 p.m.	27	0%
08:06:22 p.m.	28	4%
08:07:24 p.m.	27	0%
08:08:26 p.m.	27	0%
08:09:28 p.m.	27	0%
08:10:30 p.m.	28	4%
08:11:31 p.m.	27	0%
08:12:34 p.m.	27	0%
08:13:36 p.m.	28	4%
08:14:37 p.m.	27	0%
08:15:39 p.m.	27	0%
08:17:44 p.m.	28	4%
08:18:46 p.m.	27	0%
08:19:47 p.m.	27	0%
08:20:49 p.m.	28	4%
08:21:51 p.m.	28	4%
08:22:54 p.m.	27	0%
08:24:58 p.m.	27	0%
08:28:05 p.m.	27	0%

Hora	Distancia medida	% error
07:23:09 p.m.	25	0%
07:24:10 p.m.	25	0%
07:25:12 p.m.	25	0%
07:26:14 p.m.	25	0%
07:27:16 p.m.	25	0%
07:28:17 p.m.	25	0%
07:29:19 p.m.	25	0%
07:30:22 p.m.	25	0%
07:31:23 p.m.	25	0%
07:32:25 p.m.	25	0%
07:33:27 p.m.	25	0%
07:34:28 p.m.	25	0%
07:35:30 p.m.	25	0%
07:36:32 p.m.	25	0%
07:37:33 p.m.	25	0%
07:38:35 p.m.	25	0%
07:39:38 p.m.	25	0%
07:40:39 p.m.	25	0%
07:41:41 p.m.	25	0%
07:42:43 p.m.	25	0%
07:43:45 p.m.	25	0%
07:44:46 p.m.	25	0%
07:45:48 p.m.	25	0%
07:46:50 p.m.	25	0%
07:47:52 p.m.	25	0%
07:48:53 p.m.	25	0%
07:49:55 p.m.	25	0%
07:50:57 p.m.	26	4%
07:51:58 p.m.	25	0%
07:53:00 p.m.	25	0%
07:54:02 p.m.	25	0%
07:55:03 p.m.	25	0%
07:56:05 p.m.	25	0%
07:57:07 p.m.	25	0%
07:58:09 p.m.	25	0%
07:59:10 p.m.	25	0%
08:00:12 p.m.	25	0%
08:01:14 p.m.	25	0%
08:02:15 p.m.	25	0%
08:03:17 p.m.	25	0%
08:04:19 p.m.	25	0%
08:05:21 p.m.	25	0%
08:06:22 p.m.	25	0%
08:07:24 p.m.	25	0%
08:08:26 p.m.	25	0%
08:09:28 p.m.	25	0%
08:10:30 p.m.	25	0%
08:11:31 p.m.	25	0%
08:12:34 p.m.	25	0%
08:13:36 p.m.	25	0%
08:14:37 p.m.	25	0%
08:15:39 p.m.	25	0%
08:17:44 p.m.	25	0%
08:18:46 p.m.	25	0%
08:19:47 p.m.	25	0%
08:20:49 p.m.	25	0%
08:21:51 p.m.	25	0%
08:22:54 p.m.	25	0%
08:24:58 p.m.	25	0%
08:28:05 p.m.	25	0%